НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ПРИКЛАДНОЇ МАТЕМАТИКИ

Кафедра системного програмування і спеціалізованих комп‘ютерних систем

«До захисту допущено»

Завідувач кафедри

_____ Тарасенко В.П.
(підпис) (ініціали, прізвище)

“___” червня 2020 р.

Дипломний проект

на здобуття ступеня бакалавра

з напряму підготовки **6.050102 «Комп'ютерна інженерія»**

на тему: «Університетський портал звітності»

Виконав: студент IV курсу, групи КВ-61

Тельман Валентин Вікторович
(підпис)

Керівник: доц. каф. СПіСКС, к. т. н. Зорін Ю.М.
(підпис)

Консультант з нормоконтролю, доц. каф. СПіСКС, к.т.н. Клятченко Я.М. _____
(підпис)

Рецензент _____
(посада, науковий ступінь, вчене звання, науковий ступінь, прізвище та ініціали) (підпис)

Засвідчую, що у цьому дипломному проекті немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2020 року

Національний технічний університет України
“Київський політехнічний інститут
імені Ігоря Сікорського”

Факультет прикладної математики

Кафедра системного програмування і спеціалізованих комп’ютерних систем

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки: **123 «Комп’ютерна інженерія»**

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ В. О. Романкевич
“___” _____ 2020 р.

ЗАВДАННЯ
на дипломний проект студенту
Тельману Валентину Вікторовичу

1. Тема проекту: «Університетський портал звітності»,
   керівник проекту доц. каф. СПСКС, к.т.н. Зорін Ю. М.,
   затверджені наказом по університету від «___» травня 2020 р. № _____.
2. Термін подання студентом проекту: «___» травня 2020 р
3. Вихідні дані до проекту:
   - університетський веб додаток.
4. Зміст пояснювальної записки:
   - аналіз існуючих освітніх веб порталів та обґрунтування теми дипломного проекту;
   - аналіз технологій для розробки університетського веб порталу;
   - структура проекту. Опис роботи модулів;
   - інтерфейс користувача.

5. Перелік графічного матеріалу:

- розташування модулів програми (схема структурна);
- загальна схема проекту (схема структурна);
- загальна схема бази даних (схема структурна);
- основний маршрут користувача (схема алгоритму).

6. Консультанти:

Питання	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Клятченко Я.М., к.т.н., доцент каф. СПіСКС		

7. Дата видачі завдання: “30” жовтня 2019 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів виконання дипломної Роботи	Термін виконання етапів роботи	Примітка
1.	Вивчення літератури за тематикою проекту	18.11.2019	
2.	Розроблення та узгодження технічного завдання	30.11.2019	
3.	Аналіз існуючих рішень	10.01.2020	
4.	Підготовка матеріалів першого розділу дипломного проекту	18.01.2020	
5.	Підготовка матеріалів другого розділу дипломного проекту	01.04.2020	
6.	Підготовка графічної частини дипломного проекту	01.05.2020	
7.	Оформлення документації дипломного проекту	14.05.2020	
8.	Попередній огляд матеріалів диплому на кафедрі	20.05.2020	

Студент _____ Тельман В. В.

Керівник проекту _____ Зорін Ю. М.

АНОТАЦІЯ

Кваліфікаційна робота включає пояснювальну записку (50 с., 33 рис., 4 додатки).

Об’єкт розробки – створення університетського веб порталу для покращення процесу навчання шляхом надання інструментів для взаємодії між викладачами та студентами.

Розроблена програма дозволяє:

- приймати, оброблювати та надсилати на сервер дані та дії користувача;
- зберігати інформацію в базі даних та отримувати їх при необхідності;
- виконувати процес взаємодії між викладачем та студентом.

В ході виконання дипломного проекту:

- розроблено веб портал для покращення процесу навчання;
- проведено аналіз існуючих рішень;
- розроблено інтерфейс користувача, структуру сервера та бази даних. Налагоджено їх взаємодію.

Ключові слова: ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ВЕБ ДОДАТОК, ІНТЕРФЕЙС КОРИСТУВАЧА, СЕРВЕР, БАЗА ДАНИХ, C#, HTML, CSS, JAVASCRIPT, ASP.NET MVC, SQL.

ABSTRACT

Qualification work includes an explanatory note (50 pages, 33 pictures, 4 appendices).

The object of development is creating a university web portal to improve the learning process by providing tools for interaction between teachers and students.

The developed program allows:

- receive, process and send to the server data and user actions;
- store information in a database and receive them if required;
- perform the process of interaction between teacher and student.

During the implementation of the diploma project:

- developed a web portal to improve the learning process;
- conducted the analysis of existing decisions;
- developed user interface, server structure and database. Their interaction has been established.

Keywords: SOFTWARE, WEB APPLICATION, USER INTERFACE, SERVER, DATABASE, C#, HTML, CSS, JAVASCRIPT, ASP.NET MVC, SQL.

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
			Документація загальна			
			Новорозроблена			
	А4	ІАЛЦ.467200.002 ТЗ	Університетський портал	4		
			звітності			
			Технічне завдання			
	А4	ІАЛЦ.467200.003 ТП	Університетський портал	2		
			звітності			
			Відомість технічного			
			проекту			
	А4	ІАЛЦ.467200.004 ПЗ	Університетський портал	50		
			звітності			
			Пояснювальна записка			
	А4	ІАЛЦ.467200.005 Д1	Університетський портал	1		
			звітності			
			Розташування модулів			
			програми			
			Схема структурна			

Зм	Лист	№ докум.	Підп	Дата	ІАЛЦ.467200.001 ОА		
Розроб.		Тельман В.В.			Університетський портал звітності.	Літ.	Лист 1 / Листів 2
Перев.		Зорін Ю.М.					
Н. контр.		Клятченко Я.М.				КПІ ім. Ігоря Сікорського, ФПМ, КВ-61	
Затв.		Тарасенко В.П.			Опис альбому		

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.467200.006 Д2	Університетський портал	1		
			звітності			
			Загальна схема проекту			
			Схема структурна			
	А4	ІАЛЦ.467200.007 Д3	Університетський портал	1		
			звітності			
			Загальна схема бази даних			
			Схема структурна			
	А4	ІАЛЦ.467200.008 Д4	Університетський портал			
			звітності	1		
			Основний маршрут			
			користувача			
			Схема алгоритму			
		Диск CD-ROM	Текст ПЗ. Текст програми.	1		
			Графічний матеріал.			

Змін.	Арк.	№ докум.	Підпис	Дата	ІАЛЦ.467200.001 ОА	Арк. 2

ЗМІСТ

НАЙМЕНУВАННЯ ТА ГАЛУЗЬ РОЗРОБКИ

Назва розробки: «Університетський портал звітності».

Галузь застосування: Web-додатки, клієнти для доступу до Internet-ресурсів.

ПІДСТАВА ДЛЯ РОЗРОБКИ

Підставою для розробки є завдання на виконання роботи першого (бакалаврського) рівня вищої освіти, затверджене кафедрою системного програмування і спеціалізованих комп'ютерних систем Національного технічного університету України «Київський Політехнічний Інститут імені Ігоря Сікорського».

МЕТА І ПРИЗНАЧЕННЯ РОБОТИ

Метою даного проекту є створення веб порталу з можливістю подальшого розширення, на якому зібрані всі необхідні для навчання інструменти та ресурси такі як: розклад занять, новини, учбові матеріали тощо.

ДЖЕРЕЛА РОЗРОБКИ

Джерелом інформації є навчальні та наукові матеріали такі як підручники, статті, реферати, монографії у друкованому вигляді або ж розміщені у мережі Інтернет.

ТЕХНІЧНІ ВИМОГИ

Вимоги до програмного продукту, що розробляється

- Система авторизації користувачів.
- Спроможність працювати одночасно з великою кількістю осіб.
- Можливість подальшого розширення порталу.
- Моніторинг та запис можливих помилок і збоїв.

Вимоги до апаратного забезпечення

- Процесор: 4-ядерний процесор.
- Графічний адаптер: вбудована або дискретна відеокарта.
- Оперативна пам’ять: 8 Гб.
- Зовнішня пам’ять: 128 Гб.
- Наявність доступу до мережі Internet.

Вимоги до програмного та апаратного забезпечення користувача

- Процесор: 2-ядерний процесор.
- Оперативна пам’ять: 4 Гб.
- Зовнішня пам’ять: 64 Гб.
- Операційна система: Windows 10.
- Інтернет браузер: будь-який.
- Наявність доступу до мережі Internet.

ЕТАПИ РОЗРОБКИ

№ з/п	Назва етапів виконання дипломного проекту	Термін виконання етапів
1.	Вивчення літератури за тематикою проекту	18.11.2019
2.	Розроблення та узгодження технічного завдання	30.11.2019
3.	Аналіз існуючих рішень	10.01.2020
4.	Підготовка матеріалів першого розділу дипломного проекту	18.01.2020
5.	Підготовка матеріалів другого розділу дипломного проекту	01.04.2020
6.	Підготовка графічної частини дипломного проекту	01.05.2020
7.	Оформлення документації дипломного проекту	14.05.2020
8.	Попередній огляд матеріалів диплому на кафедрі	20.05.2020

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
			Документація загальна			
			Новорозроблена			
	А4	ІАЛЦ.467200.004 ПЗ	Університетський портал	50		
			звітності			
			Пояснювальна записка			
	А4	ІАЛЦ.467200.005 Д1	Університетський портал	1		
			звітності			
			Розташування модулів			
			програми			
			Схема структурна			
	А4	ІАЛЦ.467200.006 Д2	Університетський портал	1		
			звітності			
			Загальна схема проекту			
			Схема структурна			
	А4	ІАЛЦ.467200.007 Д3	Університетський портал	1		
			звітності			
			Загальна схема бази даних			
			Схема структурна			

Зм	Лист	№ докум.	Підп	Дата	ІАЛЦ.467200.003 ТП		
Розроб.		Тельман В.В.			Університетський портал звітності.	Літ.	Лист 1 / Листів 2
Перев.		Зорін Ю.М.					
Н. контр.		Клятченко Я.М.			Відомість технічного проекту	КПІ ім. Ігоря Сікорського, ФПМ, КВ-61	
Затв.		Тарасенко В.П.					

Поз.	Формат	ПОЗНАЧЕННЯ	НАЙМЕНУВАННЯ	Кількість аркушів	№ прим.	Примітки
	А4	ІАЛЦ.467200.008 Д4	Університетський портал	1		
			звітності			
			Основний маршрут			
			користувача			
			Схема алгоритму			
		Диск CD-ROM	Текст ПЗ. Текст програми.	1		
			Графічний матеріал.			

					ІАЛЦ.467200.003 ТП	Арк. 2

ЗМІСТ

ДОДАТКИ

Додаток 1. Копії графічних матеріалів

- ІАЛЦ. 467200.005 Д1. Розташування модулів програми. Схема структурна;
- ІАЛЦ. 467200.006 Д2. Загальна схема проекту. Схема структурна;
- ІАЛЦ. 467200.007 Д3. Загальна схема бази даних. Схема структурна;
- ІАЛЦ. 467200.008 Д4. Основний маршрут користувача. Схема алгоритму.

Додаток 2. Лістинг програми

Додаток 3. Презентація бакалаврського проекту

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

СУБД – система управління базами даних.

ASPX (Active server page extended file) – технологія для побудови веб-сайтів.

ASP.NET (Active server page for .NET) – платформа для розробки веб-додатків на базі .NET.

CSS (Cascade style sheets) – формальна мова опису зовнішнього вигляду документа.

Framework (фреймворк) – програмне забезпечення, яке полегшує розробку і об’єднання компонентів великого програмного проекту.

HTML (Hypertext Markup language) – мова гіпертекстової розмітки.

IDE (Integrated development environment) – інтегроване середовище розробки

JS (JavaScript) – скриптова мова програмування.

JSON (JavaScript object notation) – текстових формат обміну даними

MVC (Model view controller) – одна із популярних програмних архітектур.

.NET Framework (.NET) – фреймворк, який служить фундаментом для низки інших платформ.

ВСТУП

В наші дні Інтернет активно використовується для навчання. Дистанційне навчання стало звичним ділом, хоча декілька століть тому назад його взагалі не існувало. Лише з появою стабільного поштового зв’язку, люди змогли отримувати учбові матеріали, листуватися з викладачами та здавати екзамени. Уже на початку XX століття з появою телеграфів, телефонів і телевізорів якість навчання вийшла на новий рівень; кількість людей, що навчались, зросла в сотні разів. Але залишалась одна велика проблема: учні не мали можливості отримати зворотній зв’язок. Та вже на початку нового тисячоліття був подоланий і цей недолік.

Напевно, сьогодні не залишилось навчальних закладів, які б не мали власного інтернет-ресурсу. Більше того, деякі заклади взагалі не існують фізично, перенісши все, від подачі документів до видачі сертифікату в онлайн.

Можливість в декілька кліків дізнатись про зміни у розкладі, слідкувати за своїм рейтингом, вчасно отримувати новини, швидко завантажувати і відправляти учбові матеріали та домашні завдання, - все це є перевагами використання веб порталу в процесі навчання.

АНАЛІЗ ІСНУЮЧИХ ОСВІТНІХ ВЕБ ПОРТАЛІВ ТА ОБҐРУНТУВАННЯ ТЕМИ ДИПЛОМНОГО ПРОЕКТУ

1.1 Види освітніх веб порталів, їх особливості

Навчальні заклади використовують Інтернет в різних масштабах. Одні мають у своєму розпорядженні малий сайт із загальною інформацією про свою установу; інші не існують фізично, пропонуючи усі послуги в режимі онлайн.

Усі портали можна розділити по декільком признакам:

- За рівнем доступу:
  - Відкритий - відвідати сайт може будь-який із користувачів мережі Інтернет;
  - Закритий - доступ мають обмежене коло людей. Часто такий ресурс розміщений у внутрішньому домені навчального закладу, тобто звичайна людина навіть не зможе побачити такий сайт. Також існує варіант введення системи авторизації.
- За основним призначенням:
  - Інформаційний – носить довідковий характер. Може містити в собі історію виникнення; перелік факультетів, кафедр, спеціальностей, предметів, викладачів; вимоги до кандидатів; список заслуг; матеріально-технічне забезпечення тощо;
  - Корпоративний – несе ціль спрощувати та покращувати процес навчання як викладачам, так і студентам. Може містити в собі віртуальні бібліотеки; інструменти комунікації між педагогами та учнями; засоби для проведення занять, здач екзаменів.
- За типом навчального закладу:
  - Дошкільний – ясла, дитячі садки;

- Середні – школи, гімназії, ліцеї;

- Професійно-технічні – професійно-технічне училище, професійний ліцей;
- Вищі – коледж, інститут, університет

Необхідно зазначити, що якщо портал носить винятково довідковий характер, то він є відкритим. І навпаки: частіше за все, ресурс не залишають доступним для всіх користувачів, якщо він був розроблений виключно для осіб, що мають відношення до певного учбового закладу.

Також зі збільшенням рівня освітньої установи ростуть і вимоги у використанні Інтернету. Якщо дитячий садку зовсім не обов'язково мати свою сторінку в мережі, то вже важко уявити існування, наприклад, інституту без багатофункціонального веб порталу.

Часто вищі навчальні заклади мають одразу два сайти, тим самим одночасно і допомагаючи абітурієнтам, і покращуючи умови навчання педагогам і учням.

Під час написання даної роботи було розроблено варіант корпоративного університетського інтернет-сайту закритого типу.

1.2 Аналіз існуючих інтернет-ресурсів

В цьому підрозділі розглянуто декілька прикладів веб порталів навчальних закладів. Вони відрізняються між собою видом, складністю, представленими можливостями та цільовою аудиторією.

Почнемо з розгляду простого шкільного сайту. Він створений за допомогою конструктора сайтів uCoz, якому вже 14 років, має простий вигляд (рис 1.1) та малий функціонал, що характерно для більшості освітніх установ подібного рівня.

Рисунок 1.1 – Головна сторінка шкільного сайту

Незважаючи на систему авторизації, за рівнем доступу даний сайт можна віднести до відкритого. Будь-який користувач може знайти його в мережі та подивитися інформацію на будь-якій вкладці (наприклад, оцінки школярів з різних предметів)

Щодо призначення даного ресурсу, то воно є виключно інформаційним. Тут присутня інформація як і загального характеру, так і спеціального.

Розглядаючи технічну сторону, то, як було сказано раніше, портал створений за допомогою конструктора сайтів, в якому немає нічого цікавого та функціональності якого навряд чи буде достатньо для потреб вищих навчальних закладів.

Перейдемо до аналізу інтернет-ресурсів НТУУ «КПІ ім. Ігоря Сікорського». Почнемо зі сторінки Факультету Прикладної Математики (ФПМ) (рисунок 1.2).

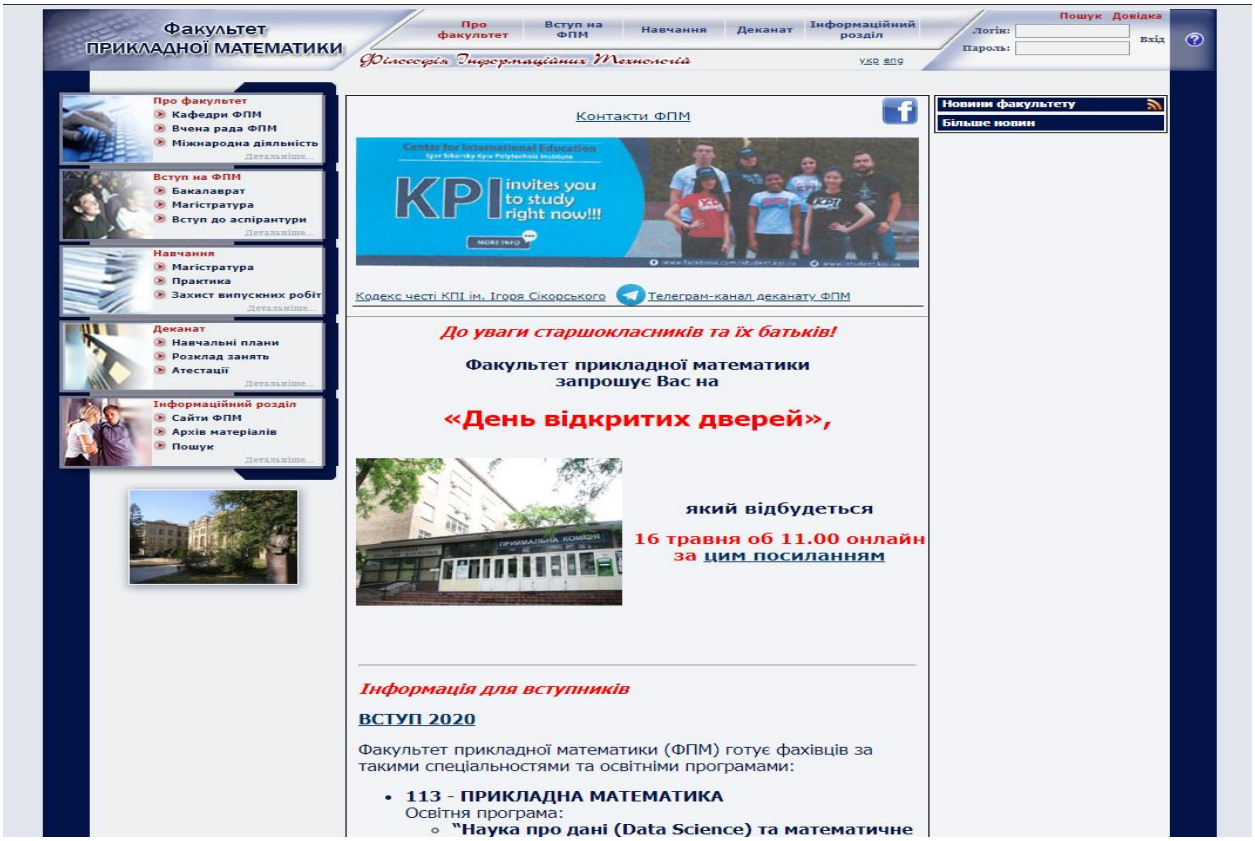


Рисунок 1.2 – головна сторінка ФПМ

За рівнем доступу даний інтернет-сайт є відкритим: усі відвідувачі можуть отримати інформацію з будь-якої сторінки, завантажувати учбові матеріали, рейтинг студентів тощо. Система авторизації необхідна тільки для додавання та зміни файлів і видачі дозволів.

Основне призначення ресурсу – інформаційне. Тут можна знайти всю необхідну інформацію, що стосується факультету. Також варто відмітити наявність архіву матеріалу, який покращує умови навчання.

На відміну від попереднього, даний сайт розроблений з використанням мов програмування. Для створення візуальної (фронтенд) частини

застосовані HTML, CSS, JS; для серверної (бекенд) – програмний компонент Java – Java servlet, що на сьогодні є застарілим. Яка при цьому була залучена СУБД – з’ясувати не вдалось.

Розглянемо ще один корисний онлайн-ресурс – rozklad.kpi.ua (рисунок 1.3).

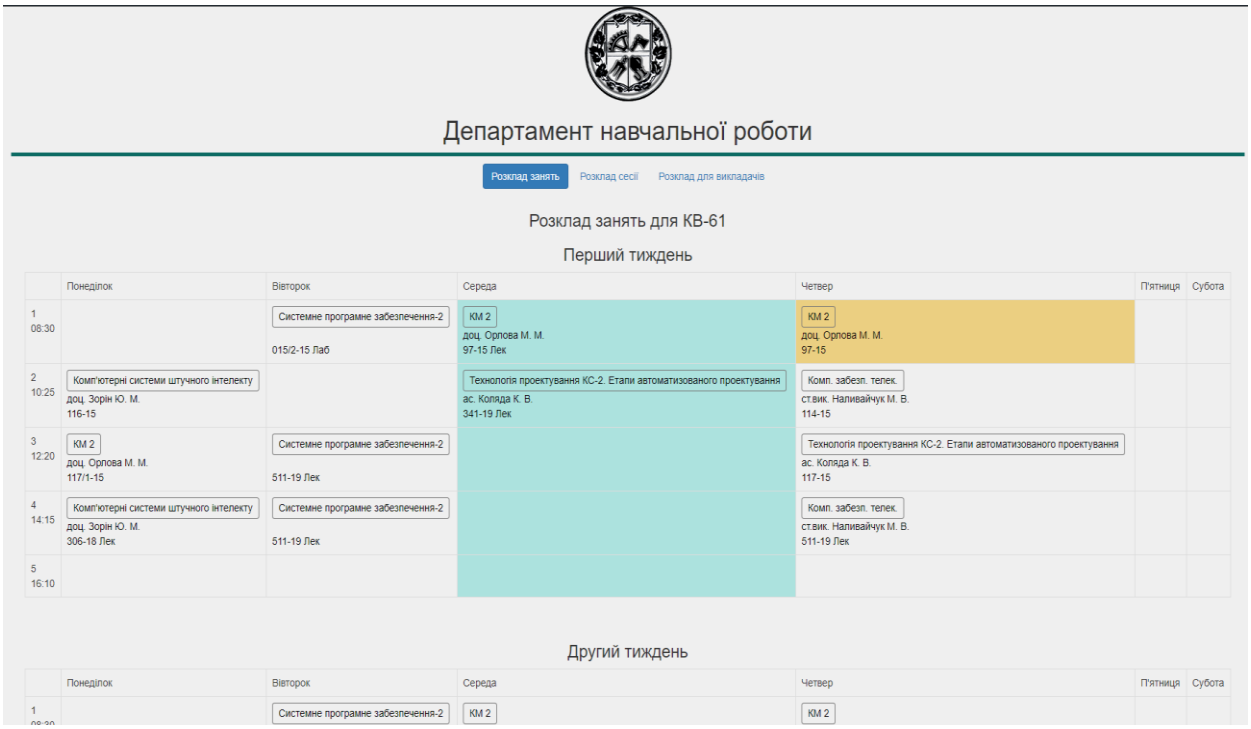


Рисунок 1.3 – головна сторінка сайту rozklad.kpi.ua

Він, як всі розглянуті в даному розділі портали, є відкритого типу. Системи авторизації немає, але це й не потрібно для головної мети сайту – відображення розкладу занять та сесії як студентам, так і викладачам. Виходячи з цього, за основним призначенням, портал можна віднести до корпоративного типу (хоча він і є вузьконаправленим).

Щодо технічної сторони додатку, то за візуальну частину відповідають HTML, CSS, JS – дуже ефективна і популярна зв’язка на сьогоднішній день.

Для написання серверної сторони використано сервіс ASPX платформи .NET Framework.

1.3 Обґрунтування теми дипломного проекту

Для обґрунтування теми дипломного проекту зазначимо недоліки розглянутих у попередньому підрозділі порталів:

- Застарілість технологій.

  Недаремно засоби розробки постійно оновлюються, деякі взагалі втрачають свою актуальність, а на зміну їм приходять нові. Покращені версії пропонують ширші можливості при менших затратах. Для створення дипломного проекту була використана одна із передових технологій.

- Мала кількість розширень і допоміжних інструментів для розробки.

  Даний пункт насамперед важливий програмістам. Будь-яка програма – це набір файлів із символами, тому теоретично є можливим її створення у звичайному текстовому редакторі та подальшою компіляцією. Але такий підхід займе непомірно багато часу навіть на простий набір тексту, не кажучи вже про пошук і виправлення помилок. Тому з метою спрощення створення програми були придумані так звані IDE та розширення до них. На кількість останніх прямо впливає популярність програмної платформи, а обрана для створення дипломного проекту займає найвищі сходинки в рейтингу поширеності. Також творці даного продукту ведуть пряму підтримку написання допоміжних інструментів.

- Проблеми з підтримкою.

Здебільшого недостатньо просто створити та віддати додаток на користування – потрібно слідкувати за ним і надалі. Інколи виникають виняткові ситуації, що не були виявлені при тестуванні. Також може виникнути проблема з розширенням функціоналу. Для цього необхідно мати гнучку архітектуру, яку і має обрана для створення ресурсу платформа розробки.

- Невисока навантажувальна здатність.

  Треба зазначити, що даний пункт є критичним тільки для додатків з високим обігом користувачів або на яких проходять події з великим напливом людей. Для прикладу розглянемо онлайн-реєстрацію на зовнішнє незалежне оцінювання студентами одного із навчальних закладів. У певний день відкривається подача заявок і сотні студентів перебувають на сайті одночасно. У кращому випадку будуть затримки, та частіше ресурс просто вийде з ладу. Дану проблему можна вирішити масштабуванням додатку, а обрана платформа розробки вже має вбудованими необхідні засоби.
- Розкиданість інтернет-ресурсів в межах одного навчального закладу.

  Даний пункт стосується зручності використання. В деяких навчальних закладах розклад занять знаходиться на одному сайті, новини – на іншому. Один викладач спілкується та передає учбові матеріали через пошту, другий – через власний блог, третій – через месенджер. В даній роботі зроблена спроба зібрати в одне місце основні засоби для навчання.

АНАЛІЗ ТЕХНОЛОГІЙ ДЛЯ РОЗРОБКИ УНІВЕРСИТЕТСЬКОГО ВЕБ ПОРТАЛУ

2.1 Програмні засоби для створення веб додатків. Причини використання фреймворків

Програмні засоби для створення веб додатків можна розділити на 2 частини: програмне забезпечення та фреймворки. До першої категорії відносяться самостійні інструменти, з допомогою яких уже можна створювати продукт: мови програмування, мови розмітки, мови опису зовнішнього виду документу. Фреймворки складаються із елементів першої категорії, об’єднаних однією архітектурою; вони є каркасом системи.

В наш час більшість задач вирішується саме завдяки фреймворкам. Для обґрунтування важливості, зазначимо їх переваги:

- Помірна шаблонізація процесу розробки.

Програмні платформи створюються для виконання завдань певного класу, тому в їх основі уже лежать деякі необхідні функції, які можна одразу використовувати, при цьому можна писати свої власні. Це золота середина між використанням: мови програмування і написанням всього з нуля; конструктора сайтів, який сильно обмежує можливості програміста набором готових методів без можливості розширення.

- Зручність використання готових бібліотек.

Бібліотекою (пакетом) називають сукупність об’єктів чи підпрограм, що використовуються для розробки програмного забезпечення. Переважно їх пишуть самі програмісти, тим самим ще більше розширюючи можливості платформи. Головним плюсом пакетів є те, що вони не є обов’язковими, тому продукт не буде перенаповнений непотрібним функціоналом.

- Використання єдиної архітектури.

При написанні програми розробник фактично обмежений лише можливостями та синтаксисом мови програмування, тому одне і те саме рішення може виглядати по-різному. У проектах, над якими працюють групи людей, це може сильно заважати злагодженій праці. Використання єдиного підходу дозволяє вирішує дану проблему. Навіть новому співробітнику буде просто розібратись зі структурою продукту.

Вище зазначені переваги, що притаманні усім фреймворкам. Про плюси та мінуси окремих буде написано в наступних підрозділах.

2.2 Варіанти фреймворків

Для аналізу були обрані наступні фреймворки:

- AngularJS
- Laravel
- ASP.NET MVC 5

Основним критерієм відбору став рівень популярності. Також перечисленні кандидати мають спільну архітектуру – MVC, з неї і почнемо. MVC – один із популярних нині архітектурних шаблонів (паттернів), що використовується у розробці веб-додатків (рисунок 2.1).

Основний принцип роботи полягає у розподілі даних додатку (Model, модель), керуючого елементу (Controller, контро́ллер) та інтерфейсу користувача (View, представлення) на три окремих компоненти так, щоб їх можна було змінювати незалежно один від одного.

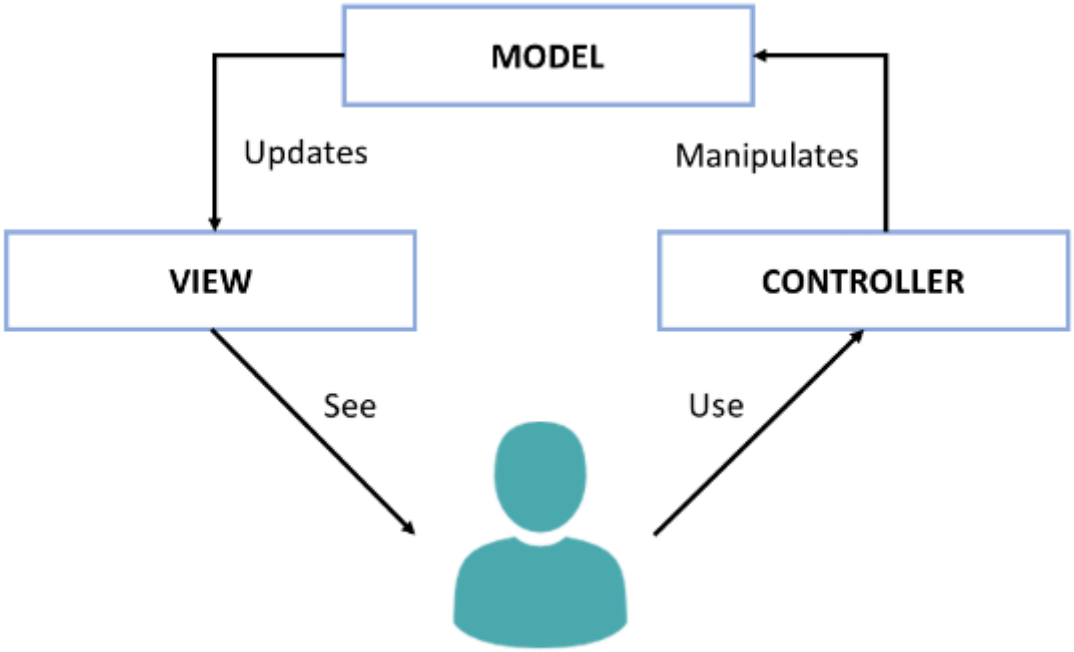


Рисунок 2.1 – паттерн MVC

При цьому:

1. Модель – займається обробкою даних та містить логіку додатку. Самі дані зазвичай зберігаються в базі даних.
2. Контролер – ідентифікує дії користувача та викликає відповідні ресурси.
3. Представлення – відповідає за представлення усіх даних користувачу, реагуючи на зміні в моделі.

Необхідно зазначити, що існують різні модифікації даного шаблону такі як: HMVC, MVVM, MVP та інші. Вони мають схожу структуру, лише

замінюючи один із компонентів або ж додаючи новий. Детально на них зупинятися не будемо.

MVC не має строгої реалізації, тому, наприклад, бізнес-логіка може зберігатись як і в моделі, так і в контролері. Також часто додаток складається з декількох шарів (рисунок 2.2). Кінцеві деталі реалізації установлюють фреймворки.

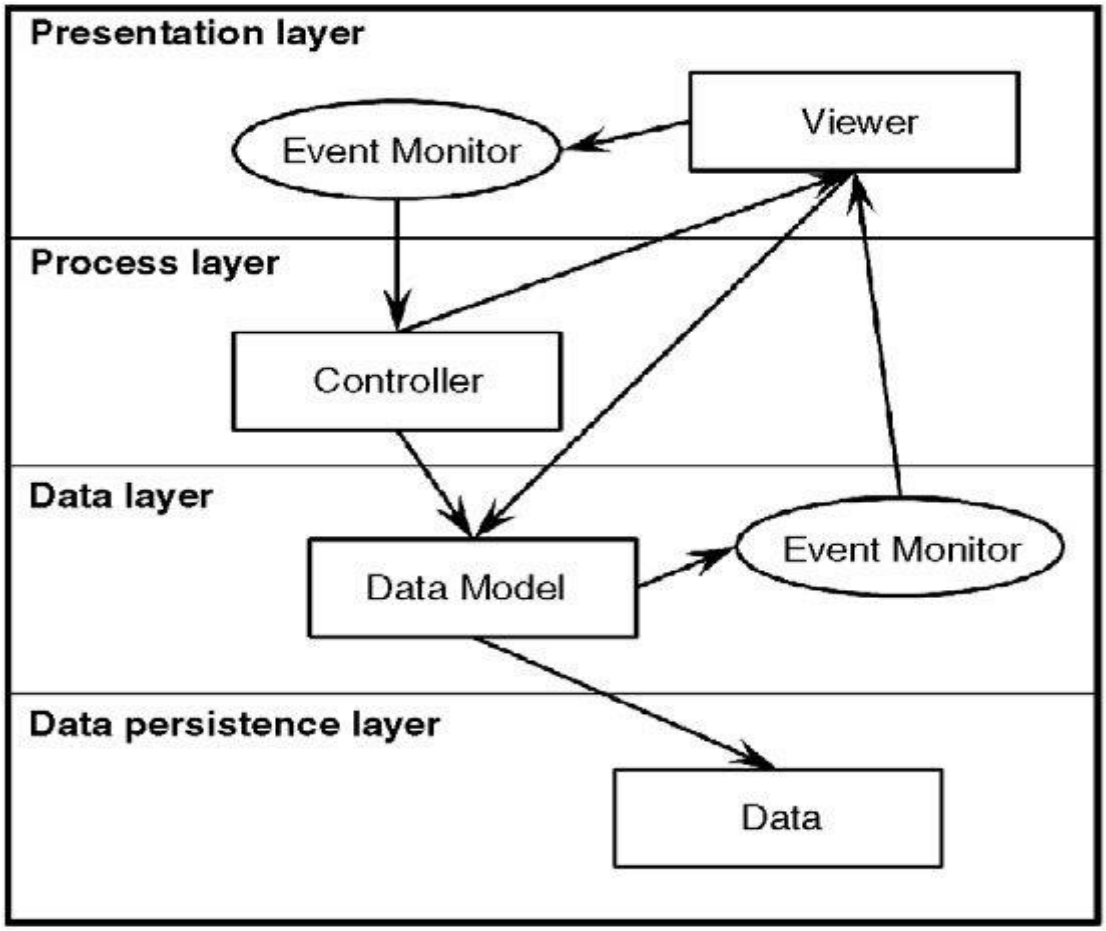


Рисунок 2.2 – Багаторівневий MVC

Даний архітектурний шаблон недаремно є таким популярним серед розробників веб-додатків, адже він має такі переваги:

- Чіткий розподіл зобов’язань.

Головний плюс патерну. Мова йдеться про розбиття додатку на інтерфейс користувача (зовнішній вигляд) та бізнес-логіку (функціонал), що значно спрощує структуру програми.

Над окремими частинами можуть працювати різні спеціалісти, використовуючи різні мови програмування.

- Зручність розробки для різних пристроїв.

Часто веб-сайти створюються одразу під декілька видів пристроїв (комп'ютер, планшет, смартфон). Застосування MVC дозволяє створити представлення для кожного з них, використовуючи при цьому лише одну модель.

- Зручність тестування.

Так як кожен компонент додатку не залежить від іншого, то його можна тестувати окремо від інших, що значно простіше. Такий тип перевірки має назву «Модульне тестування».

Перейдемо до аналізу першого фреймворку – AngularJS [1]. Відноситься до класу фронтенд-розробки. В його основі лежить мова програмування JavaScript. Також взаємодіє з HTML, розширюючи його. В результаті цього з'являється можливість створювати динамічні види на веб порталах Для обміну даними використовується формат JSON. Приклад структури програми зображений на рисунку 2.3.

Основне призначення – розробка односторінкових додатків – сайтів, що використовують єдиний HTML-документ в якості обгортки для інших сторінок. При цьому усе базується на використанні таких компонентів:
MVC фреймворк, HTML5, маршрутизація між представленнями на рівні браузеру, шаблонізатор, AJAX.

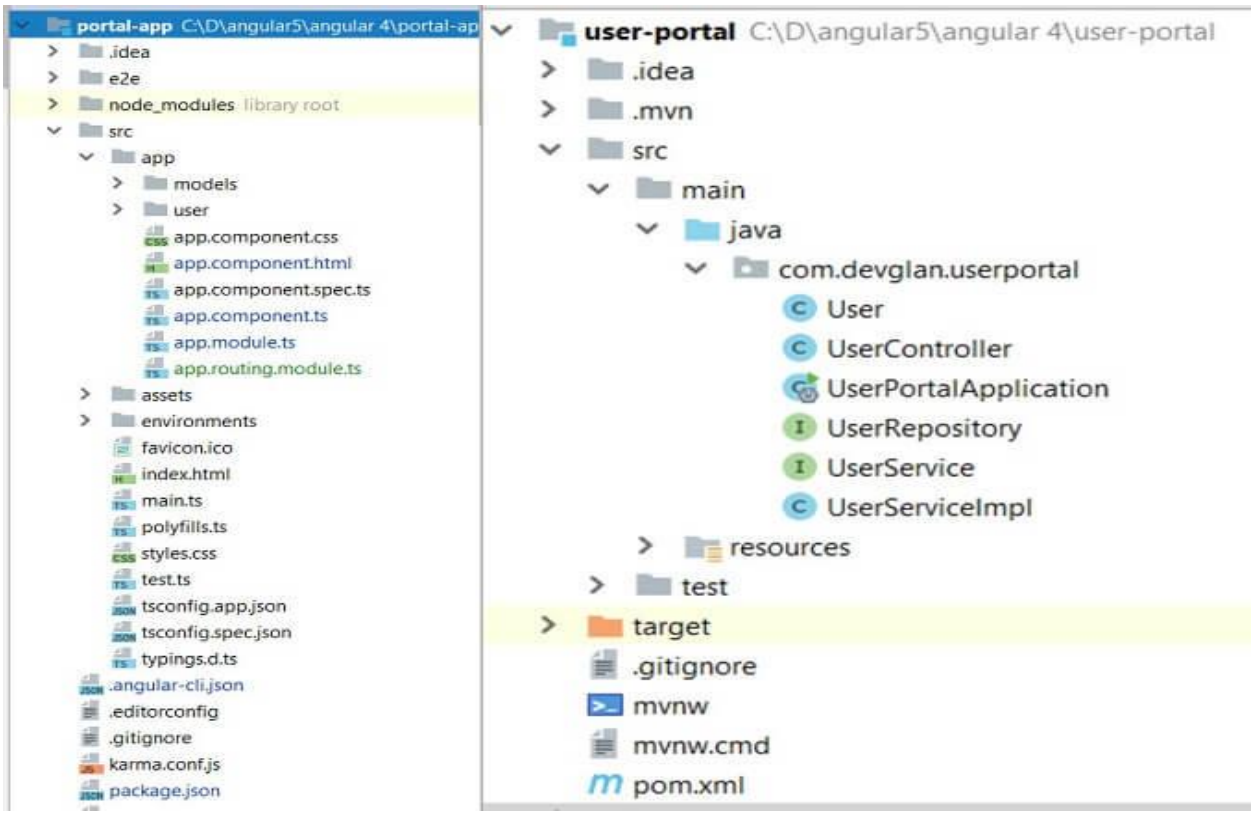


Рисунок 2.3 – Структура Angular-додатку

Розглянемо плюси та мінуси даного фреймворку:

+ Використання класичного патерну MVC, що передає всі плюси останнього.
+ Розширення синтаксису HTML, що призводить до підвищення зручності написання коду та його чистоти.
+ Краще інших підходить для розробки односторінкових додатків.
+ Має чітко визначену структуру, завдяки якій новим користувачам буде легше увійти в процес розробки.
- Не підходить для розробки крупних проектів, так як має обмеження при роботі з великим об’ємами даних та медіа-контенту.
- Чітка структура платформи є мінусом для тих розробників, хто потребує максимальної гнучкості при створенні продукту.

Можна зробити висновок, що AngularJS краще всього підходить для розробки невеликих проектів. Але для комплексного університетського порталу він не підходить.

Перейдемо до розгляду наступного фреймворку – Laravel [2]. Він займає найвищу сходинку по популярності серед тих, що базуються на мові програмування PHP. Відноситься до класу бекенд-розробки, але має підтримку підключення до JS бібліотеки vue.js, що дозволяє паралельно працювати над інтерфейсом користувача. Підходить для розробки додатків будь-якої складності. Приклад структури програми зображений на рисунку 2.4.

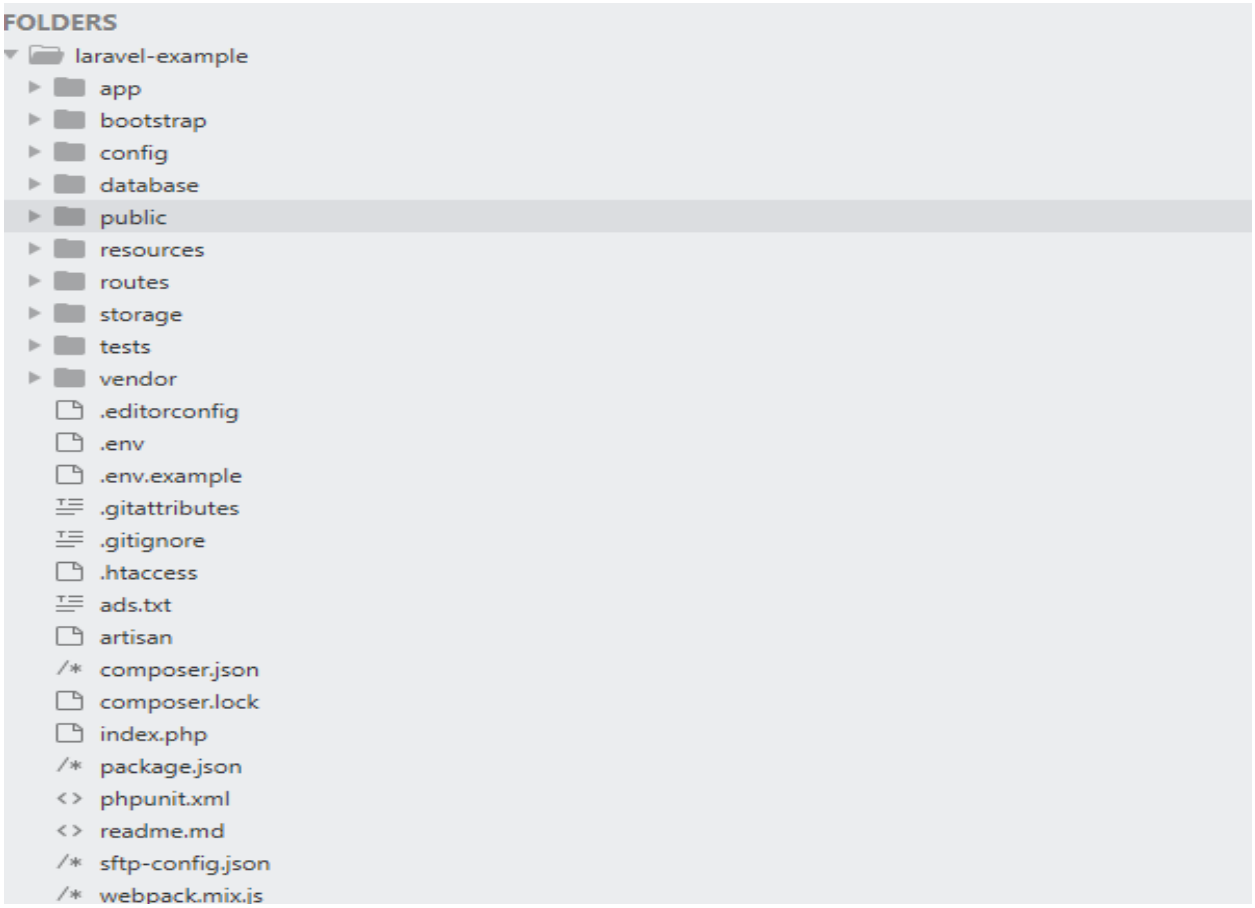


Рисунок 2.4 – Структура Laravel-додатку

Розглянемо плюси та мінуси даного фреймворку:

+ Великий обсяг зрозумілої документації (на англійській мові).
+ Через високу популярність платформи, створена об’ємна база уроків, статей, курсів, книг.
+ Дозволяє створювати додатки будь-якої складності.
+ Значно спрощує і оптимізує базову мову PHP.
+ Наявність власного шаблонізатору Blade.
- Програмісти, які не володіють хоча б базовим рівнем англійської мови, втратять суттєву частину навчальних матеріалів.
- Через значне спрощення написання коду, для певних розробників буде проблемою повернутись до розробки на чистому PHP або ж просто змінити програмну платформу.

Підсумовуючи, Laravel не має значних недоліків і є гідним кандидатом для написання університетського порталу, при умові володіння програмістом мовою PHP хоча б на середньому рівні.

Останній кандидат для аналізу – фреймворк ASP.NET MVC 5 [3]. Він створений для розробки як і серверної, так і клієнтської частини додатку. В його основі лежить потужна мова програмування C#, що і відповідає за бекенд-розділ. Для налаштування фронтенду зазвичай використовується класична зв’язка HTML, CSS, JS, хоча існує можливість підключення додаткових засобів, наприклад, уже розглянутого нами AngularJS. Також система має власний шаблонізатор Razor, який дозволяє писати код на C# прямо у інтерфейсі користувача. Це значно спрощує комунікацію між моделлю і представленням.

ASP.NET MVC 5 є частиною іншого фреймворку - .NET Framework, який за роки свого існування накопичив без перебільшення гігантський об’єм

бібліотек та сервісів, які покривають майже всі потреби розробки програм. При цьому вони не є обов'язковими і підключаються по мірі необхідності.

Має декілька готових шаблонів (наприклад, з налаштованою системою авторизації), що обираються в декілька кліків на початку створення додатку. Після цього з'являється проект, що має структуру, зображену на рисунку 2.5.

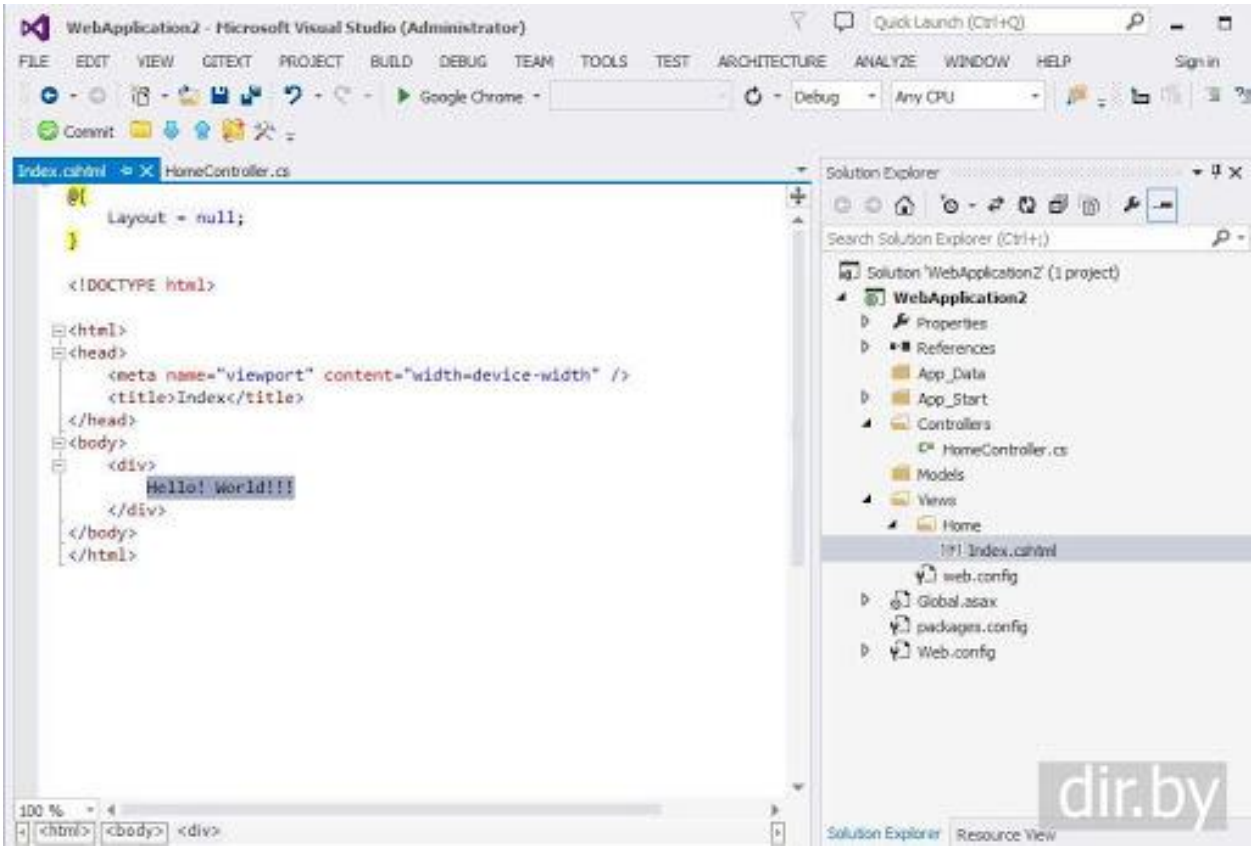


Рисунок 2.5 - Структура ASP.NET MVC 5-додатку

Розглянемо плюси і мінуси даного фреймворку:

+ Об'ємна база технічної документації та навчальних засобів як і на англійській, так і на російській мовах.
+ Величезна кількість бібліотек і сервісів, що пропонують готовий для використання функціонал.

+ Вся інфраструктура побудована на основі реалізацій інтерфейсів, що дозволяє розширити і без того немалі можливості.
+ Можливість створення додатку будь-якої складності.
+ Використання власного шаблонізатору Razor, що значно спрощує розробку.
+ Проста і потужна система тестування, яка має графічний інтерфейс, декілька видів тестів, можливість автоматичного запуску при певних змінах у коді тощо.
- Відносно високий поріг входження, через використання мови програмування C#.
- В порівнянні з більш простими фреймворками є повільнішим в роботі.

Беручи до уваги усі його плюси, ASP.NET MVC 5 добре підходить для розробки університетського веб-порталу, тому було обрано саме його.

2.3 Компоненти обраного фреймворку

Розділимо аналіз на 2 частини: інструменти розробки інтерфейсу користувача та серверу. Про особливості сховища даних буде зазначено в наступних підрозділах окремо.

Почнемо з розгляду фронтенд-частини. Для її створення використовується класична зв'язка HTML, CSS, JS.

HTML – мова гіпертекстової розмітки, яка використовується практично у всіх браузерах. Вона і є скелетом того, що бачить користувач. Складається з так званих тегів, які містять атрибути, що задають певні властивості останнім. Структура кожного HTML-документу містить у собі початковий тег <html>, всередині якого лежать ще два: <head> (голова) і <body> (тіло). В першому зберігається технічна інформація, яка необхідна для браузера: назва, автор сторінки, а також, наприклад, підключені CSS стилі. У другому

розміщується все те, що буде бачити користувач. Приклад зображений на рисунку 2.6.

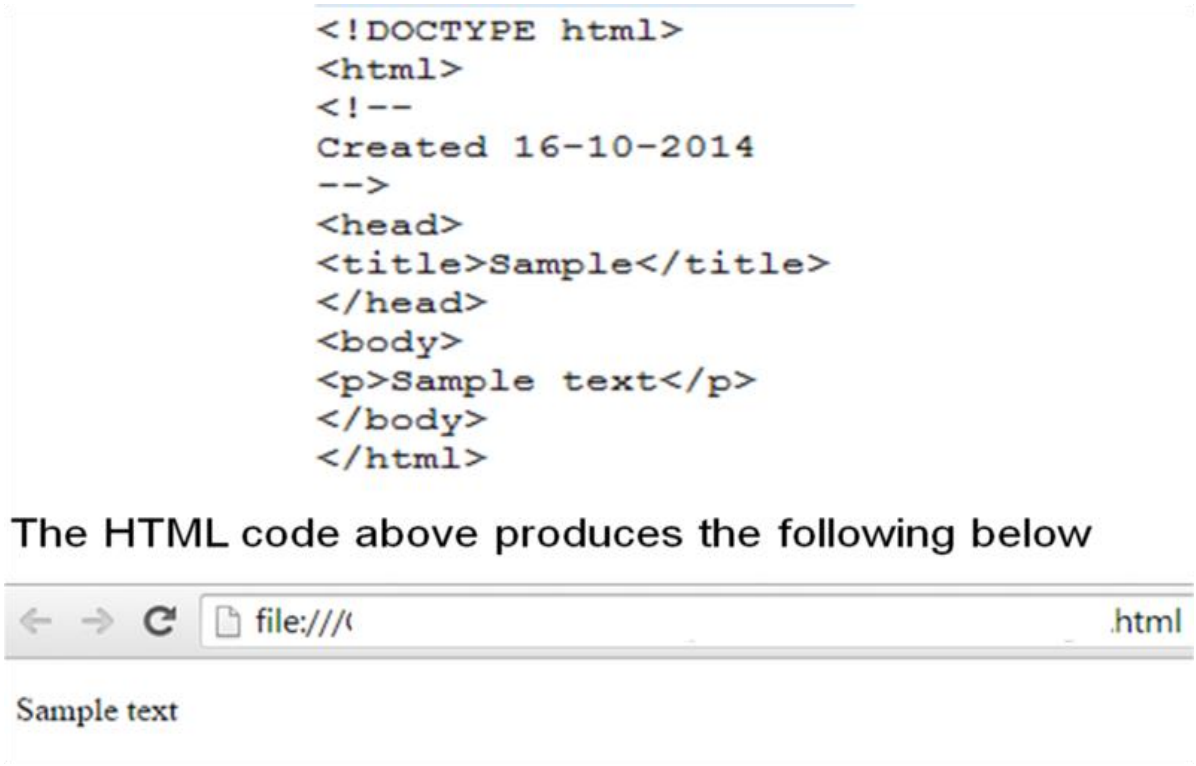


Рисунок 2.6 – Приклад HTML-документу

Основна мета мови – вивід статичної веб-сторінки на екран, яка не має ні стилю, ні анімацій, ні можливості взаємодії. Для оформлення зовнішнього виду HTML-документу використовується CSS. З його допомогою можна змінювати кольори, форми, шрифти, границі елементів, а також додавати анімації.

Складається з так званих правил: селектора и блоку оголошень. В ролі першого можуть бути теги, класи (можуть бути присвоєні декільком елементам), ідентифікатори (можуть бути присвоєні лише 1 елементу), а також їх комбінації. Другий містить список оголошень: властивість CSS і значення, що розділені двокрапкою. Першою може бути параметри об’єкту (ширина, висота, шрифт, колір); значення задаються спеціальною назвою, числом, пікселями, процентами тощо.

Після підключення файлу стилів, CSS-властивості будуть одразу застосовані до всіх відібраних елементів HTML-документу. Приклад CSS файлу зображений на рисунку 2.7.

```
h1 {
    font-family: courier, courier-new, serif;
    font-size: 20pt;
    color: blue;
    border-bottom: 2px solid blue;
}
p {
    font-family: arial, verdana, sans-serif;
    font-size: 12pt;
    color: #6B6BD7;
}
.red_txt {
    color: red;
}
```

Рисунок 2.7 – Приклад CSS-файлу

Для створення процесу взаємодії між користувачем та HTML-документом використовується мова програмування JavaScript. Існує багато модифікацій, таких як TypeScript, CoffeeScript та інші. Всі вони привносять певні покращення, та все ж це лише обгортки, код яких інтерпретується на класичний JS, тому що браузер працює саме з ним.

Основне призначення – розробка фронтенд-частини веб-додатків, але в наш час використання настільки поширилось, що дана мова програмування підходить для написання і серверів, і мобільних проектів, і навіть офісних програм.

Саме у веб-розробці переважно використовується для обробки дій користувача (натиск кнопки на клавіатурі чи миші, наведення на об’єкт, завантаження файлу тощо). Або у окрему файлі з розширенням .js і подальшим підключенням, або прямо в HTML-документі в тезі <script>

пишуться функції, які можна призначити будь-яким об’єктам сторінки за допомогою так званих подій - атрибуту, що починається з «on» (onclick, onload, onkeypress тощо). Приклад застосування JS зображено на рисунку 2.8.

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>handlebarsJS example</title>
  </head>
  <body>
  <h1>Let's say hello to:</h1>
  <div id="nameoutput">
  </div>

  <script id="person-template" type="text/x-handlebars-template">
    <p class="person">{{firstname}} {{lastname}}</p>
  </script>
  <script src="handlebars.js"></script>
  <script>
  window.onload = function () {
      var source = document.getElementById("person-template").innerHTML;
      var template = Handlebars.compile(source);
      var context = {firstname: "John", lastname: "Doe"};
      var output = template(context);
      document.getElementById("nameoutput").innerHTML = output;
  }
  </script>
  </body>
</html>
```

Рисунок 2.8 – Приклад взаємодії JS і HTML

Для покращення процесу взаємодії між моделлю та представленням, в ASP.NET MVC 5 існує власний шаблонізатор Razor (рисунок 2.9). Від дозволяє передавати дані прямо на HTML-сторінку, а також програмувати в ній за допомогою C#. Його синтаксис складається з символу @ та наступним за ним серверним кодом.

Для обґрунтування необхідності Razor, перелічимо його переваги:

- Значно спрощує інтеграцію між клієнтською та серверною частинами додатку;
- Суттєво знижує час на написання коду в представленні;
- Має дуже простий синтаксис;
- Не потребує спеціального текстового редактору для роботи;

- Підтримує так званий intellisense – помічник при написанні коду, який виводить список можливих для застосунку змінних та функцій.

```
<body>
    <div class="page">
        <div id="header">
            <div id="title">
                <h1>My MVC Application</h1>
            </div>
            <div id="logindisplay">
                @Html.Partial("_LogOnPartial")
            </div>
            <div id="menucontainer">
                <ul id="menu">
                    <li>@Html.ActionLink("Home", "Index", "Home")</li>
                    <li>@Html.ActionLink("About", "About", "Home")</li>
                </ul>
            </div>
        </div>
        <div id="main">
            @RenderBody()
            <div id="footer">
            </div>
        </div>
    </div>
</body>
```

Рисунок 2.9 – Приклад HTML-сторінки з Razor

Нарешті перейдемо до розгляду основного інструменту для розробки серверної частини додатку – мови програмування високого рівня C# [4].

Є об’єктно-орієнтованим, має C подібний синтаксис, строгу типізацію та вміщує величезну кількість можливостей, серед яких: перевантаження операторів, поліморфізм, наслідування, анонімні функції з підтримкою замикання, події, делегати, властивості, узагальнення типів та методів тощо. Насправді неможливо перелічити увесь функціонал даної мови, так як в офіційній документації знаходиться більше 30 глав. Все це робить його по-справжньому потужним інструментом в умілих руках.

За 20 років свого існування C# набув великої популярності при розробці не тільки веб, а й десктопних та мобільних додатків. За допомогою нього побудовані такі фреймворки як WinForms, WebForms, Xamarin, WPF.

Вся функціональність ASP.NET MVC, від взаємодії з базою даних, через бізнес-логіку, до інтеграції з клієнтською частиною, повністю покладається на C#. Приклад коду зображений на рисунку 2.10.

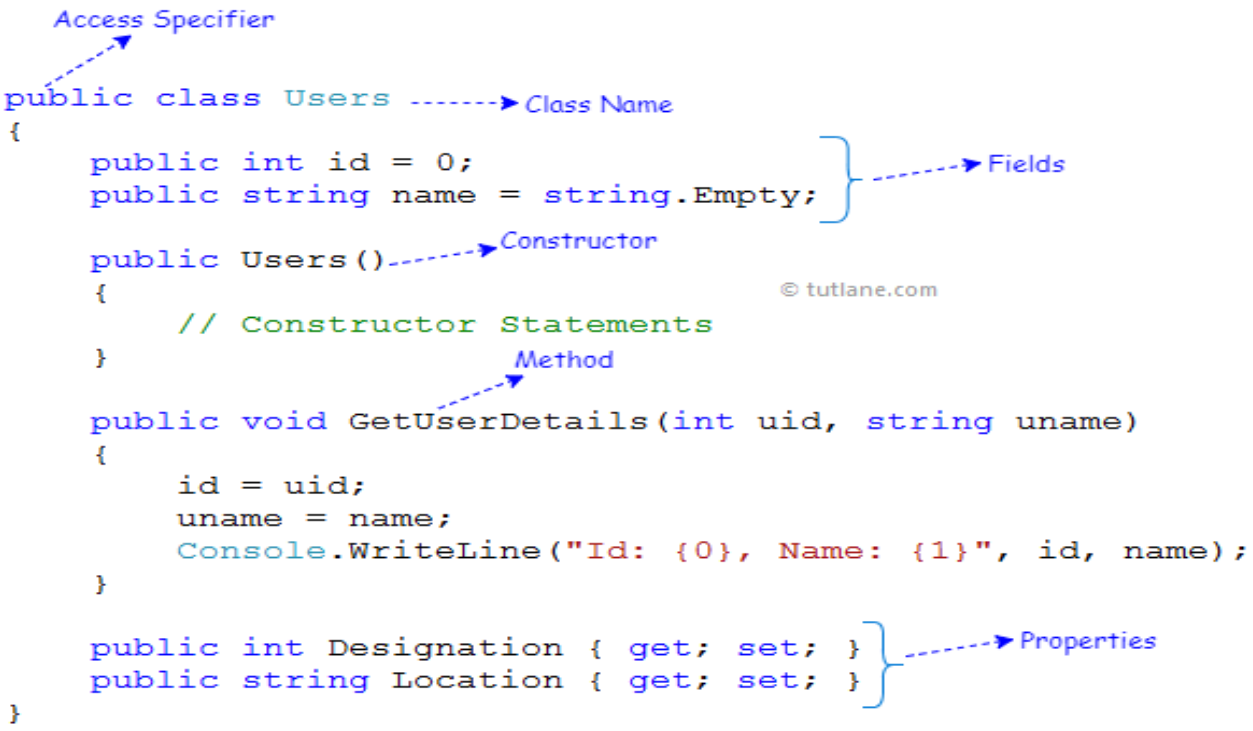


Рисунок 2.10 – Приклад C# коду

2.4 Вибір бази даних

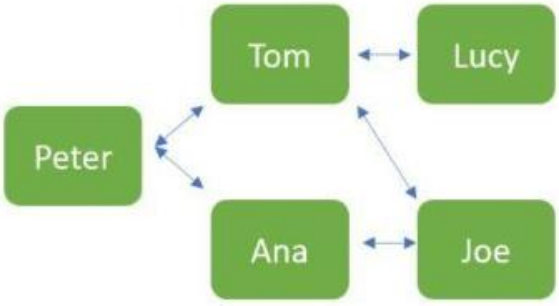
Бази даних можна поділити на дві великі групи: реляційні (SQL) та нереляційні (NoSQL). Їх різниця полягає у структурі, а також у протилежних підходах до зберігання даних.

Представники першої групи відзначаються своєю структурованістю та більше підходять для зберігання даних реальних об’єктів. Наприклад, якщо створена сутність «Студент» і вона має атрибути «ФІО», «Вік», «Група», то всі її об’єкти підпорядковуються заданій схемі. Приклади реляційних СУБД: Microsoft SQL Server, SQLite, PostgreSQL.

Представники другої групи навпаки мають неструктуровані, гнучкі схеми. Інформація зберігається у вигляді ієрархічної структури даних.

Наприклад, один об’єкт сутності «Студент» може мати лише атрибути «Вік» і «Стать», інший – «ФІО», «Зріст», «Колір очей».

На рисунку 2.11 показані різні підходи до відображення однакових даних: реляційний (a) та нереляційний (b).



(a)

NODE	FRIENDS
Peter	{Tom, Ana}
Tom	{Joe, Lucy, Peter}
Ana	{Joe, Lucy, Peter}
Lucy	{Tom}
Joe	{Tom, Ana}

(b)

Рисунок 2.11 – Різні способи представлення даних

Для визначення найбільш вдалого варіанту для створення університетського веб-порталу, розглянемо вимоги, які можуть задовольнити дані напрямки.

Реляційні бази даних:

- Дані мають чітко визначену структуру;
- Проект потребує підвищеного рівня надійності;
- Необхідність використання комплексних запитів.

Нереляційні бази даних:

- Дані не мають чітко визначеної структури;
- Проект потребує підвищеної швидкості обробки даних;
- З ходом часу необхідно значно змінювати структуру бази даних.

Університетський портал одразу має чітку структуру сутностей: «Студент», «Викладач», «Предмет», «Група» тощо. Важко уявити, що вони

будуть значно змінені під час розробки. Також підвищена надійність є більш пріоритетною, ніж швидкість. Тому для розробки дипломного проекту однозначно краще підходить реляційна база даних.

Мова програмування, що використовується для створення, зміни та керуванням даними в реляційній базі даних – SQL (мова структурованих запитів) (рисунок 2.12). Перелік базових операцій, що може бути здійснений нею:

- Створення таблиці та модифікація її структури;
- Вставка нових даних (записів) у таблицю;
- Отримання даних з одної або декількох таблиць;
- Зміна записів;
- Видалення записів.

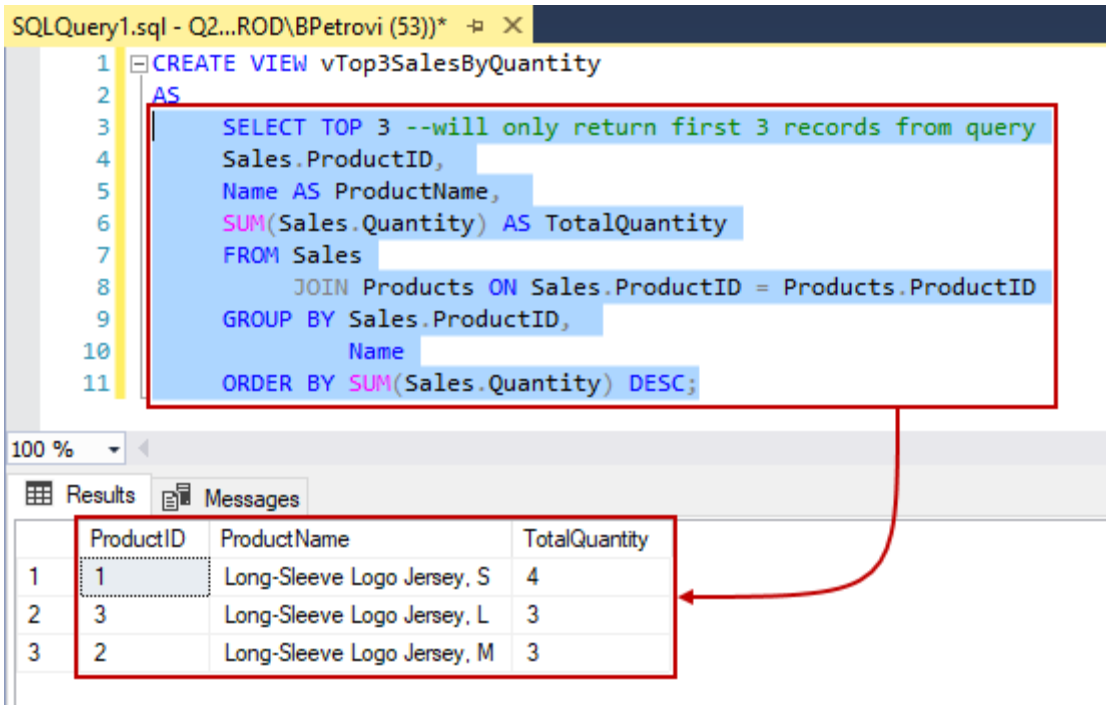


Рисунок 2.12 – Приклад SQL-запиту

Також SQL має в наявності деякі конструкції, що притаманні більшості класичним мовам програмування: команди розгалуження, цикли, складні

оператори тощо. Також є можливим управління так званими збереженими об’єктами : представленнями, тригерами, індексами, процедурами та функціями. Розглянемо їх детальніше.

Представлення (Views) використовуються для спрощення складних SQL-запитів. Реалізовані вони у вигляді віртуальних таблиць і, по суті, є фільтрами для звичайних таблиць. Також можна об’єднати декілька таблиць в одне представлення, при цьому створивши нові поля з комбінації вже існуючих (рисунок 2.13). Це значно спростить подальшу роботу з даними та скоротить об’єм коду.

```
1  CREATE VIEW [dbo].v_Student
2      AS SELECT
3          Student.Id
4          ,Student.GroupId
5          ,LastName + ' ' + FirstName + ' ' + MiddleName FIO
6          ,LastName + ' ' + LEFT(FirstName, 1) + '. ' + LEFT(MiddleName, 1) + '.' ShortFIO
7          ,LastName + ' ' + FirstName + ' ' + MiddleName + ' (' + g.Name + ')' FIOWithGroup
8      FROM Student
9      JOIN [Group] g ON g.Id = GroupId
```

Рисунок 2.13 – Приклад використання представлення

Тригери використовують для виконання певної процедури при виникненні заздалегідь встановленої події (наприклад, внесення, модифікація, видалення даних). Скажімо, є необхідним вести облік записів в таблиці. Для цього можна створити процедуру, яка буде приймати порядкових номер запису, тип дії та логін виконавця. Тоді необхідно зробити тригер, який буде спрацьовувати при будь-яких операції з даними та викликати написану процедуру. Необхідно зазначити, що є можливим її виконання як до, так і після певної дії.

Індекси є об’єктами бази даних, що використовуються для підвищення швидкості вибірки даних із таблиць. Є надзвичайно ефективними при роботі з великими об’ємами даних.

Поділяються на два типи: кластерні та некластерні. Першим типом, по суті, є сама таблиця, що відсортована по певному полю – первинному ключу.

Тобто в однієї таблиці може бути лише один такий індекс. Другий випадок являє собою спеціальну таблицю, що містить відсортовані поля таблиці, визначені розробником. При цьому самі дані в такій структурі не зберігаються, а замість них використовуються ідентифікатори. Виходить, що індексів даного типу може бути більше одного.

Перша ознака необхідності використання індексу – наявність великого об’єму даних, що часто піддаються вибірці. Для прикладу, звичайний лінійний пошук 1 необхідного запису з 1000000 в середньому займе 500000 операцій. При використанні індексу буде виконано бінарний пошук, що зменшить кількість операцій всього до 19!

Та вже ж вони мають один значний недолік: сповільнення всіх інших операцій з даними, окрім вибірки. Це відбувається через те, що індексам необхідно перебудовуватися після кожного оновлення записів. Тому їх не варто використовувати без необхідності.

Процедури є набором SQL-запитів, що виконуються в певній послідовності. Можуть приймати та повертати значення через параметри.

Їх використання несе з собою декілька переваг:

- Винесення повторюваного коду в одне місце. Такий підхід збільшує як якість, так і надійність коду.
- Забезпечення строго порядку виконання інструкцій. Навіть є можливість відмінити результати виконання попередніх дій, якщо поточна команда викликала помилку, що є зручним для тісно зв’язаних задач.

- Підвищення швидкості здійснення команд. В процедурах SQL-запити вже є скомпільованими, і тому їх залишається лише виконати.
- Запобігають SQL-ін’єкціям – процесу несанкціонованого доступу до бази даних через передачу SQL-коду через форми інтерфейсу користувача.

Функції також являють собою набір інструкцій, що виконуються в певному порядку. Але на відміну від процедур, вони повертають значення із заздалегідь вказаним типом. Воно може бути скалярним або табличним. Часто використовуються для допоміжних дій, не зв’язаних безпосередньо з маніпуляціями з даними. Простий приклад – функція, що ділить вхідну строку по вказаному символу та засовує отримані частини в результуючу таблицю (рисунок 2.14).

```
CREATE FUNCTION dbo.Split(@Str nvarchar(max), @Delimiter char(1))
RETURNS @Results TABLE (Items nvarchar(max))
AS
  BEGIN

        DECLARE @Counter int
        DECLARE @Slice nvarchar(max)
        SELECT @Counter = 1
        IF @Str IS NULL RETURN
        WHILE @Counter !=0
        BEGIN
            SELECT @Counter = CHARINDEX(@Delimiter,@Str)
            IF @Counter !=0
            SELECT @Slice = LEFT(@Str,@Counter - 1)
            ELSE
            SELECT @Slice = @Str
            INSERT INTO @Results(Items) VALUES(@Slice)
            SELECT @Str = RIGHT(@Str,LEN(@Str) - @Counter)
            IF LEN(@Str) = 0 BREAK
        END

    RETURN
END
```

Рисунок 2.14 – Приклад SQL-функції

Існують також процедурні розширення мови SQL, наприклад, Transact-SQL, що використовуються при роботі з СУБД Microsoft SQL Studio. Вони значно доповнюють функціонал за допомогою таких нововведень як: локальні та глобальні змінні, оператори керування, різні допоміжні математичні функції тощо.

2.5 Вибір інструменту для роботи з базою даних

Розглянемо 3 найпопулярніші інструменти для взаємодії обраного фреймворку та бази даних:

- ADO.NET
- Entity Framework
- Dapper

Кожен із них має особливості, переваги та недоліки. Проаналізуємо їх, щоб визначити найбільш вдалий варіант для наших цілей.

ADO.NET – технологія, яка використовується для доступу та керування даними. Основана на базі .NET Framework, тобто є стандартною. За своєю суттю є низькорівневою – програміст повинен сам налаштовувати з’єднання за допомогою так званого рядка підключення, визначати параметри та всі запити. Також цікавим є те, що поля об’єктів, повернутих з бази даних, не мають типу, тому розробник повинен самостійно вказувати тип для кожного.

Переваги даної технології:

- Швидкість роботи;
- Максимальна гнучкість налаштування.

Недоліки:

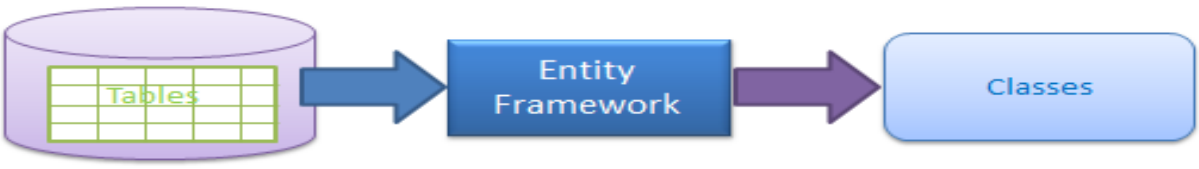
- Мінімальна кількість допоміжних засобів
- Висока складність роботи

Дану технологію доцільно використовувати лише якщо є необхідність у максимальній швидкості роботи програми та необхідності контролю усіх деталей з'єднання. Інакше висока складність та затрати часу при розробці не є виправданими.

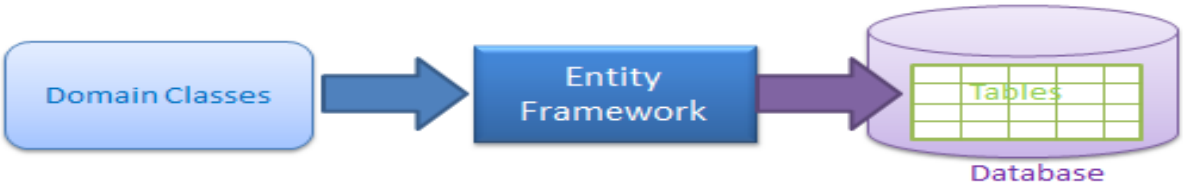
Наступна технологія для взаємодії з базою даних – Entity Framework.

Побудована на базі попередньої технології, є справні потужним інструментом, який надає широкий спектр можливостей. В її основі лежить так звана технологія об'єктно-реляційного відображення (ORM), яка пов'язує класи об'єктно-орієнтованих мов програмування та об'єкти баз даних. На виході розробник може працювати, наприклад, з останніми як із звичайними структурами даних мови програмування C#.

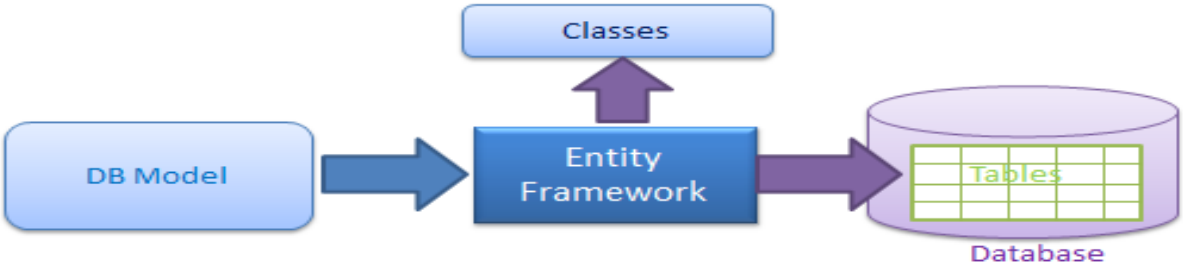
Entity Framework надає декілька можливостей для зв'язування бази даних та додатку (рисунок 2.15). Один із них - створити базу даних на основі існуючих класів. Такий підхід має назву Code First. Але існує спосіб, прямо протилежний першому – Database First. Він дозволяє згенерувати класи по вже створеній базі даних.



Generate Data Access Classes for Existing Database



Create Database from the Domain Classes



Create Database and Classes from the DB Model design

Рисунок 2.15 – Способи інтеграції з базою даних за допомогою Entity Framework

Розглянемо переваги даної технології:

- Велика кількість корисного функціоналу;
- Можливість автоматичної генерації об’єктів як бази даних, так і додатку;
- Наявність графічного інтерфейсу для роботи з об’єктами;
- Дозволяє працювати з базою даних навіть з недостатніми знаннями мови SQL.

Недоліки:

- Значне падіння швидкодії роботи з даними;
- При виникненні помилки при автогенерації коду майже неможливо її знайти і виправити;
- В цілому пропонує шаблонні рішення, що можуть не підійти для певних задач.

У підсумку можна сказати, що Entity Framework дійсно є потужним багатофункціональним інструментом для взаємодії з базою даних, який дозволить значно скоротити час на розробку проекту. Але його використання більш доцільне для насправді великих проектів. Для університетського порталу звітності дане рішення буде надмірним і краще скористатися більш простим засобом, що допоможе зберегти швидкість та надійність роботи системи.

Dapper, як і попередній кандидат, є популярним інструментом для взаємодії з базою даних, що використовує технологію ORM. Вся його функціональність зводиться до зв’язування класів програми та результатів

виконання SQL-запитів. Завдяки цьому він і є таким же швидким, як ADO.NET, але набагато зручнішим у використанні. Тому це кращий кандидат для використання у дипломному проекті.

3. СТРУКТУРА ПРОЕКТУ. ОПИС РОБОТИ МОДУЛІВ

3.1 Структура бази даних

Для зберігання інформації було обрано реляційну СУБД – Microsoft SQL Server [5].

База даних складається з 17 таблиць, декількох допоміжних представлень і процедур, 1 функції (рисунок 3.1).

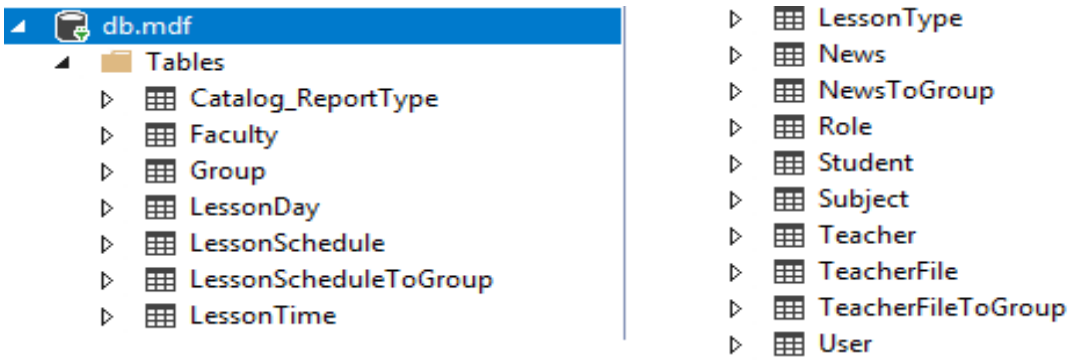


Рисунок 3.1 – Структура бази даних

Розглянемо призначення кожної:

- Catalog_ReportType – допоміжна таблиця, необхідна для отримання додатком даних для розкладу.
- Faculty – сутність типу «Факультет»;
- Group – сутність типу «Група», має зв'язок «Один-до-багатьох» з сутністю «Факультет»;
- LessonDay – сутність типу «День заняття»;

- LessonSchedule – сутність типу «Розклад заняття», має зв’язки «Один-до-багатьох» з сутностями «День заняття», «Час заняття», «Тип заняття», «Група», «Предмет», «Викладач»;

- LessonScheduleToGroup – допоміжна таблиця для реалізації зв’язку «Багато-до-багатьох» між сутностями «Розклад заняття» та «Група»;
- LessonTime – сутність типу «Час заняття»;
- LessonType – сутність типу «Тип заняття»;
- News – сутність типу «Новина», має зв'язок «Один-до-багатьох» з сутністю «Предмет»;
- NewsToGroup – допоміжна таблиця для реалізації зв’язку «Багато-до-багатьох» між сутностями «Новина» та «Група»;
- Role – сутність типу «Роль» (роль користувача на сайті);
- Student – сутність типу «Студент», має зв’язок «Один-до-одного» з сутністю «Користувач» та зв’язок «Один-до-багатьох» з сутністю «Група»;
- Subject – сутність типу «Предмет», має зв’язок «один-до-багатьох» з сутністю «Викладач»;
- Teacher – сутність типу «Викладач», має зв’язок «Один-до-одного» з сутністю «Користувач»;
- TeacherFile – сутність типу «Файл викладача», має зв’язок «один-до-багатьох» з сутністю «Викладач»;
- TeacherFileToGroup – допоміжна таблиця для реалізації зв’язку «Багато-до-багатьох» між сутностями «Файл викладача» та «Група»;
- User – сутність типу «Користувач».

Представлення використовуються для форматування даних при отриманні з бази. Наприклад, сутність «Студент» має три поля «Ім’я», «Прізвище»,

«По-батькові». Використання представлення дозволяє об’єднати ці поля, створивши нове (не змінюючи структури таблиці!) - «ФІО».

Процедури виконують схожу роль, що й функції в мовах програмування: винесення повторюваних частин запитів в одне місце. До того ж, вони є швидшими у використанні та забезпечують захист від SQL-ін’єкцій – способу несанкціонованого доступу до даних шляхом передачі шкідливого SQL-коду, наприклад, через форми введення даних в інтерфейсі користувача.

Функції відрізняються від процедур тим, що вони обов’язково повинні повертати значення. У нашому випадку використана 1 функція Split, яка приймає 2 параметри: рядок символів, та 1 символ, по якому буде розділений перший параметр. Всі отримані частини вносяться до результуючої таблиці.

3.2 Структура програми

3.2.1 Архітектура додатку

Для уникнення плутанини в термінології необхідно зазначити, що ASP.NET MVC увесь додаток носить назву «Рішення» (solution), яке в свою чергу складається з проектів (projects).

Отже, рішення веб-порталу складається з 4 проектів (рисунок 3.2), які побудовані на так званій цибулевій архітектурі (onion architecture).

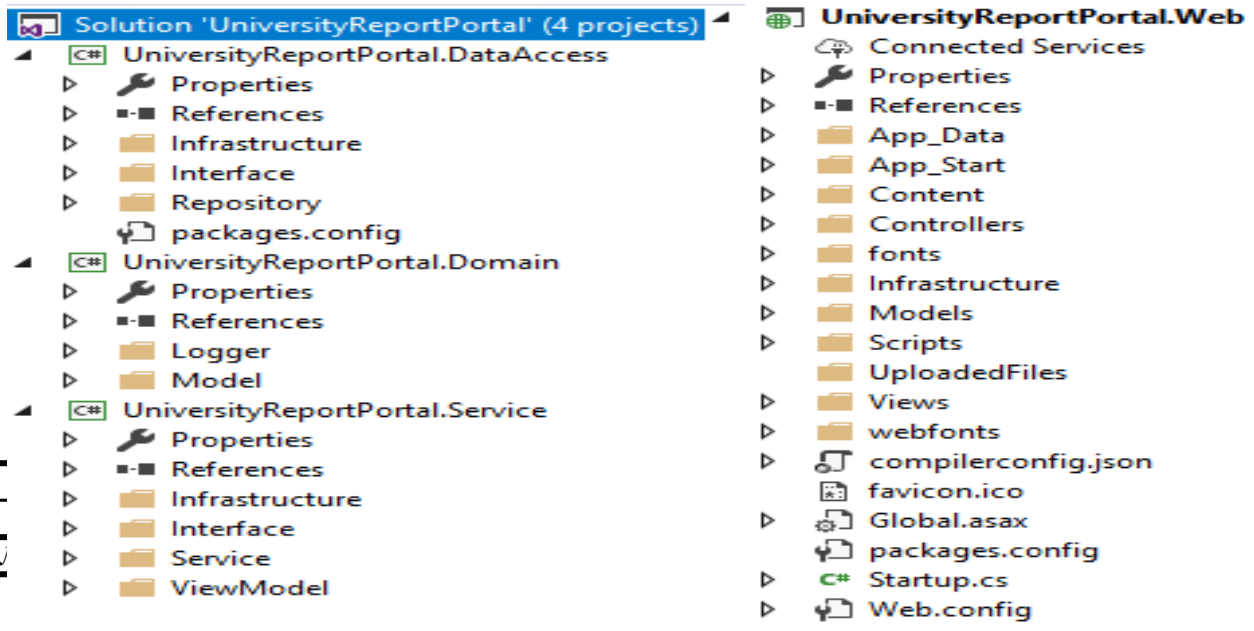


Рисунок 3.2 – Структура веб-додатку

Даний підхід добре зарекомендував себе у зв'язці з паттерном MVC; застосовується він саме до моделі останнього.

Його суть полягає у розділенні додатку на рівні. Є базовий (центральний) шар, і поверх нього створюються інші за таким правилом: кожен новий рівень може використовувати ресурси попередніх, але не може буде залежним від наступних. Тобто перший проект (ядро) є незалежним, другий – залежний від 1, але не залежний від наступних і так далі.

Використання цибулевої архітектури дозволяє чітко структурувати код, значно спрощуючи його сприйняття, тестування і підтримку.

В нашому випадку рішення розбите на 4 проекти, кожен з яких варто проаналізувати.

3.2.2 Домен

Ядром системи виступає проект з назвою Domain. Його єдина задача – зберігання класів, які використовуються у всьому рішенні. Серед них – усі моделі таблиць бази даних, а також декілька допоміжних (наприклад OperationResult, який повертає результат виконання функції і повідомлення при помилці). При цьому вони не містять ніякої логіки.

3.2.3 Модель доступу до даних

Наступним рівнем виступає проект з назвою Repository. Його задача полягає у взаємодії з SQL Server, а саме: отримання, додавання, модифікація, видалення даних, а також виклик процедур.

Починаючи з репозиторію, вводиться такий принцип об'єктно-орієнтованого програмування як інверсія управління (Inversion of control) [6]. Класичний підхід полягає у створенні об'єктів класів на нижніх рівнях

додатку і отримання до них доступу на верхніх. При цьому якщо в подальшому необхідно буде змінити реалізацію класу, то існує великий шанс зламати структуру додатку.

Інверсію управління можна реалізувати декількома способами, та в дипломному проекті це зроблено за допомогою ін'єкції залежності (Dependency injection) (рисунок 3.3).

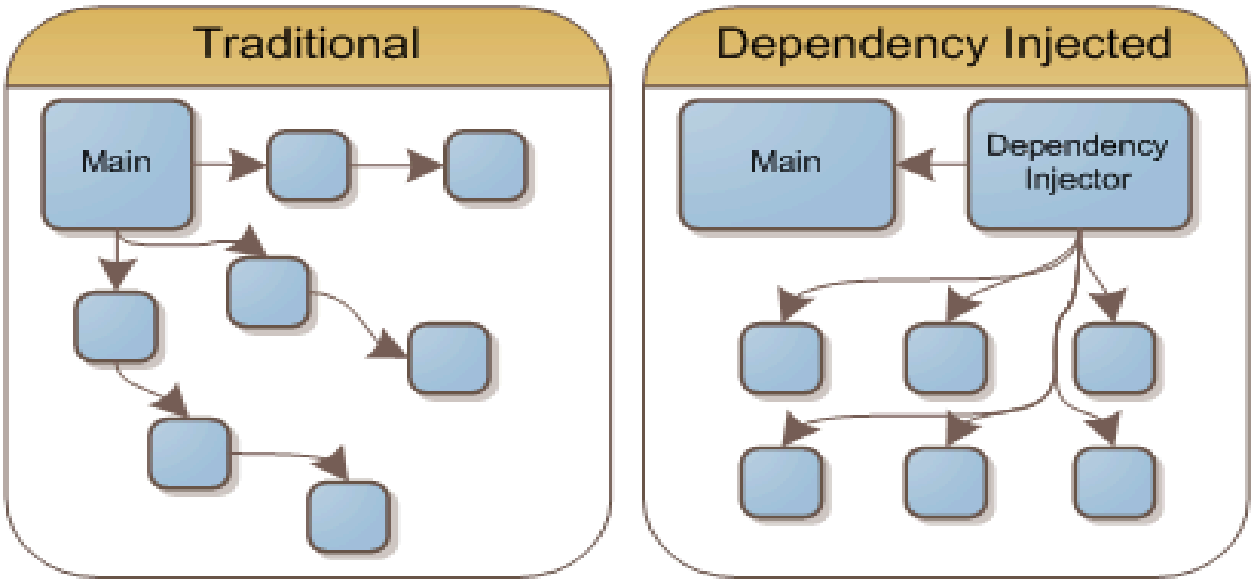


Рисунок 3.3 – Ін'єкція залежності

Суть полягає у створенні інтерфейсів для кожного класу, який містить логіку, та їх передачі через параметри конструктора усіх залежним від них сервісам. Так як інтерфейс є лише обгорткою без реалізації, то існує можливість легкої зміни реалізації функціоналу. Наприклад, клас Repository має метод Save, що приймає певний об'єкт та зберігає його у одну із таблиць SQL Server. Створивши інтерфейс IRepository з такою ж фунцією, ми передаємо через параметри конструктора саме його. Після цього приймається рішення про зміну бази даних, наприклад, MySQL, що звісно має деякі відмінності при роботі з нею. Для цього ми створюємо новий клас Repository2, що також реалізує інтерфейс IRepository, та програмуємо нову

логіку. Після тестування старий клас видаляється, а на його місце ставиться новий. В результаті ніякий залежний від даного сервісу компонент не помітив підміни.

Для спрощення процесу ін'єкції використовуються так звані IoC-контейнери. У нашому випадку застосований Ninject dependency resolver, який у спеціальному класі на найвищому рівні додатку дозволяє зв'язати реалізацію з інтерфейсом. Після цього він автоматично буде передавати в залежні сервіси реальні об'єкти. В результаті ін'єкція залежності керується в одному місці.

3.2.4 Сервісний модуль

Третій рівень має назву Service і є останнім шаром моделі в архітектурі MVC. Його суть полягає у перетворенні даних, отриманих як з бази даних, так і від користувача, тому що їх структура часто відрізняється. Наприклад, під час реєстрації користувача просять двічі ввести пароль, але в базі воно зберігається лише в одному полі. Також даний проект вміщує усю логіку.

3.2.5 Модуль взаємодії з користувачем

Верхній рівень рішення має назву Web. Містить реалізацію представлення та контроллера в архітектурі MVC. За допомогою першого взаємодіє з користувачем, оброблюючи його дії і виводячи відповідну інформацію на екран; другий - виконує роль маршрутизатору. Розглянемо детальніше його структуру та проаналізуємо найважливіші частини (рисунок 3.4).

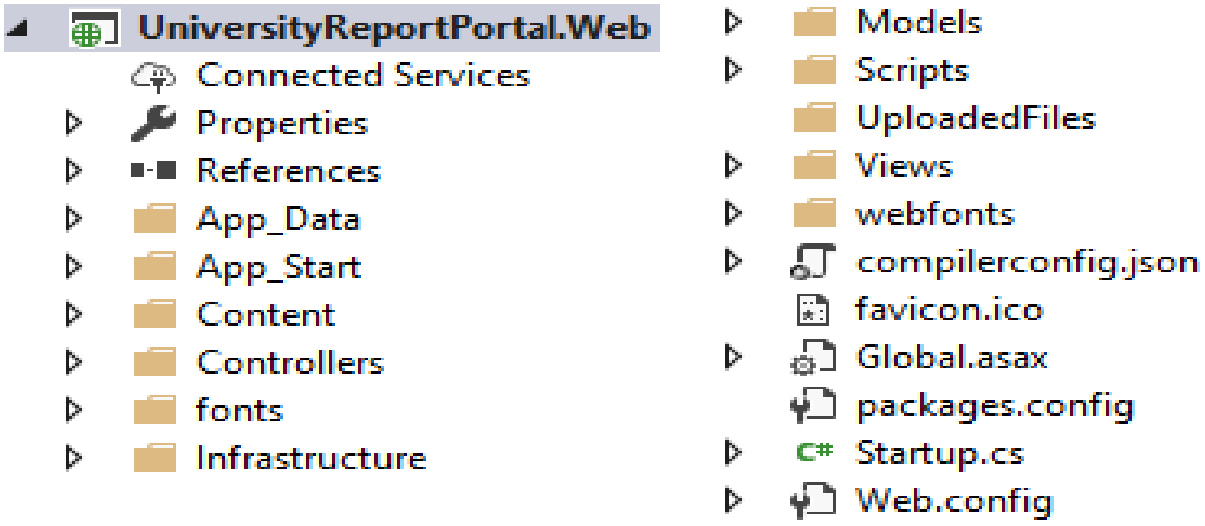


Рисунок 3.4 – Структура проекту найвищого рівня

- App_Data – Містить файл бази даних;
- App_Start – Містить класи з ініціалізацією налаштувань додатку (наприклад, початкова сторінка);
- Content – Містить файли мови CSS;
- Controllers – Містить контроллер архітектури MVC, що представлений у вигляді окремих класів.
- Infrastructure – Містить локальні сервіси (наприклад, систему авторизації та IoC-контейнер);
- Models – містить моделі представлення (не плутати з класами проекту Domain);
- Scripts – містить файли мови JS;
- UploadedFiles – каталог для зберігання файлів вчителів;
- Views – містить представлення архітектури MVC, що представлені у вигляді файлів з розширенням .cshtml (Razor + HTML);
- Інші каталоги та файли створені в процесі автогенерації рішення та виконують технічну роль.

4. ІНТЕРФЕЙС КОРИСТУВАЧА

4.1 Інтерфейс користувача

4.1.1 Початок роботи з додатком

На порталі реалізована можливість взаємодії вчителя зі студентами, а точніше певною групою.

Усі необхідні таблиці (факультети, групи, предмети тощо) заповнені при створенні додатки, пізніше адміністратор бази даних може додати нові. Створюються аккаунти для вчителів і студентів, після чого логіни та паролі видаються їх власникам.

При вході усіх користувачів зустрічає форма авторизації (рисунок 4.1).

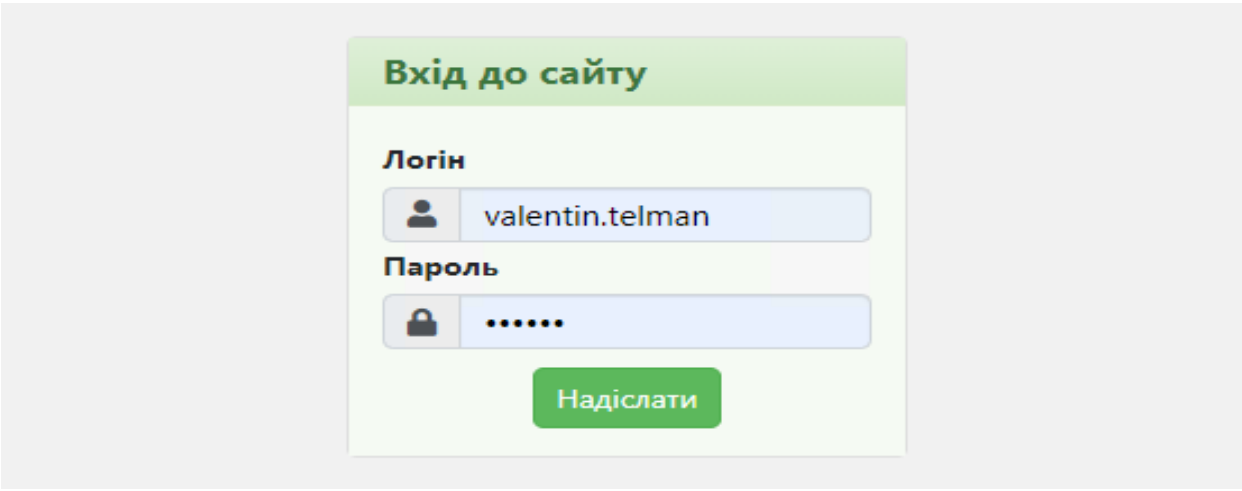


Рисунок 4.1 – Форма авторизації

4.1.2 Сторінка студента

Почнемо розгляд зі сторони студента. Після входу його зустрічає сторінка, зображена на рисунку 4.2. Верхня панель містить два пункти «Студентам» та «Вчителям» (прихована від учня), а також панель з логіном

та роллю користувача в правій частині, при натиску на яку та подальшим підтвердженням дії відбувається вихід із сайту.

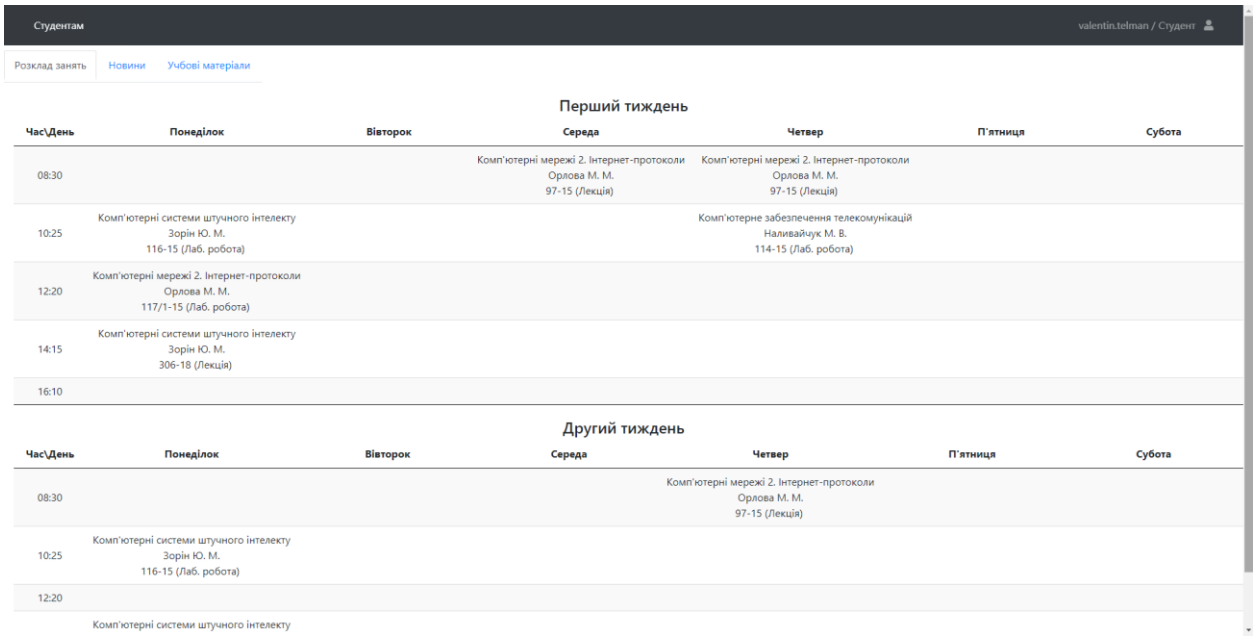


Рисунок 4.2 – Головна сторінка студента

Під «головою» сайту знаходиться підменю з трьома пунктами «Розклад занять» (головна сторінка), «Новини», «Учбові матеріали».

Перша відіграє лише інформаційну роль та немає ніякої взаємодії. Відображає розклад занять тієї групи, в якій знаходиться студент.

Перейдемо до наступної вкладки (рисунок 4.3). Вона містить новини, які були відправлені будь-яким вчителем, для групи, в якій знаходиться студент. Кожна новина складається з дати відправлення, назви, предмету. При натиску на кнопку «*Читати*», можна переглянути її вміст та закрити при натиску на хрестик в правому верхньому кутку (рисунок 4.4).

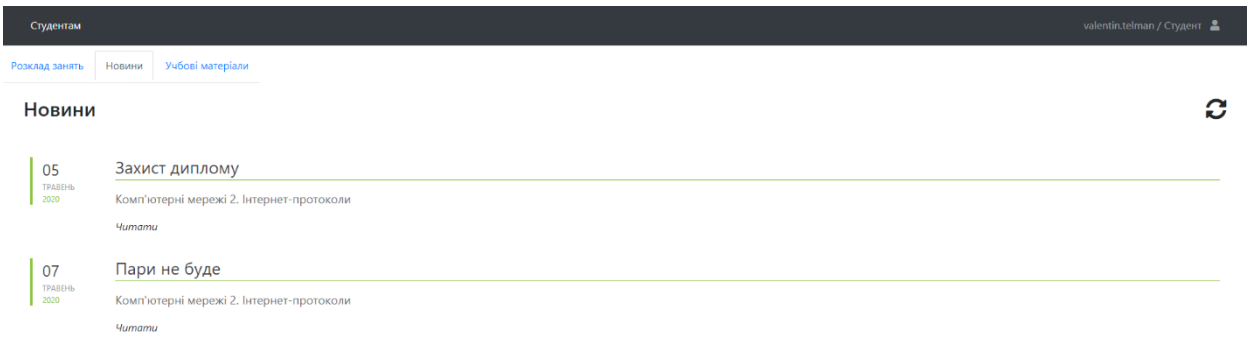


Рисунок 4.3 – Вкладка «Новини»

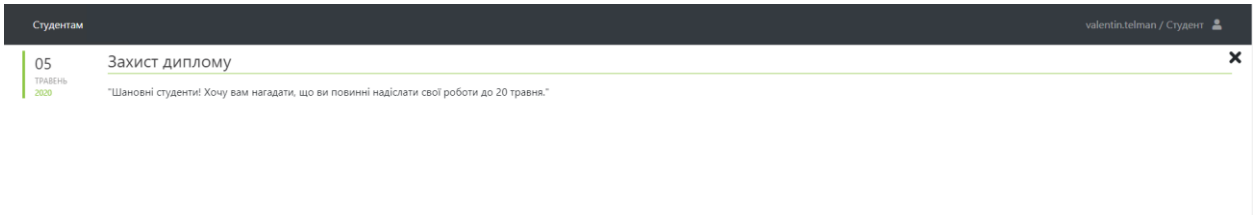


Рисунок 4.4 – Перегляд вмісту новини

Вкладка «Учбові матеріали» містить таблицю з файлами, що попередньо завантажив викладач, для групи, в якій знаходиться студент. Кожен файл має свою назву, ПІО викладача, дату створення, а також кнопку завантаження (рисунок 4.5).

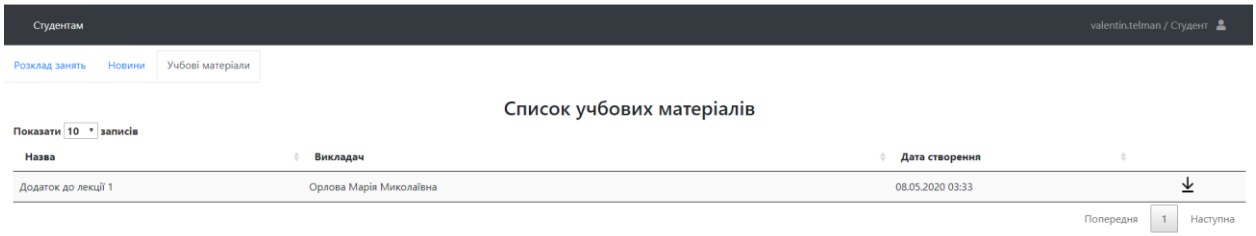


Рисунок 4.5 – Вкладка «Учбові матеріали»

4.1.3 Сторінка викладача

Тепер перейдемо до розгляду додатку зі сторони викладача. Після успішної авторизації відкривається сторінка, зображена на рисунку 4.6.

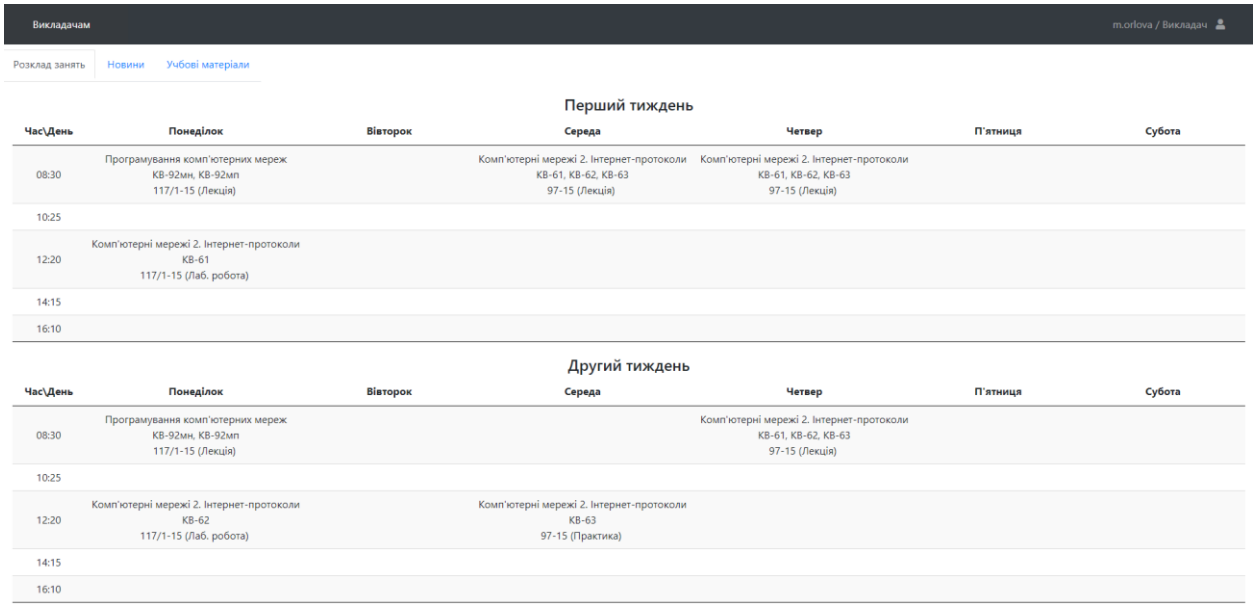


Рисунок 4.6 – Головна сторінка викладача

Структура «голови» сайту ідентична, але функціонал вкладок відрізняється, то ж почнемо з першої. Тут виводиться розклад занять для авторизованого викладача. Він може бути змінений за допомогою подвійного кліку на одну із клітинок, і в залежності від того чи було на цьому місці заняття, відкриється форма створення чи редагування (рисунок 4.7).

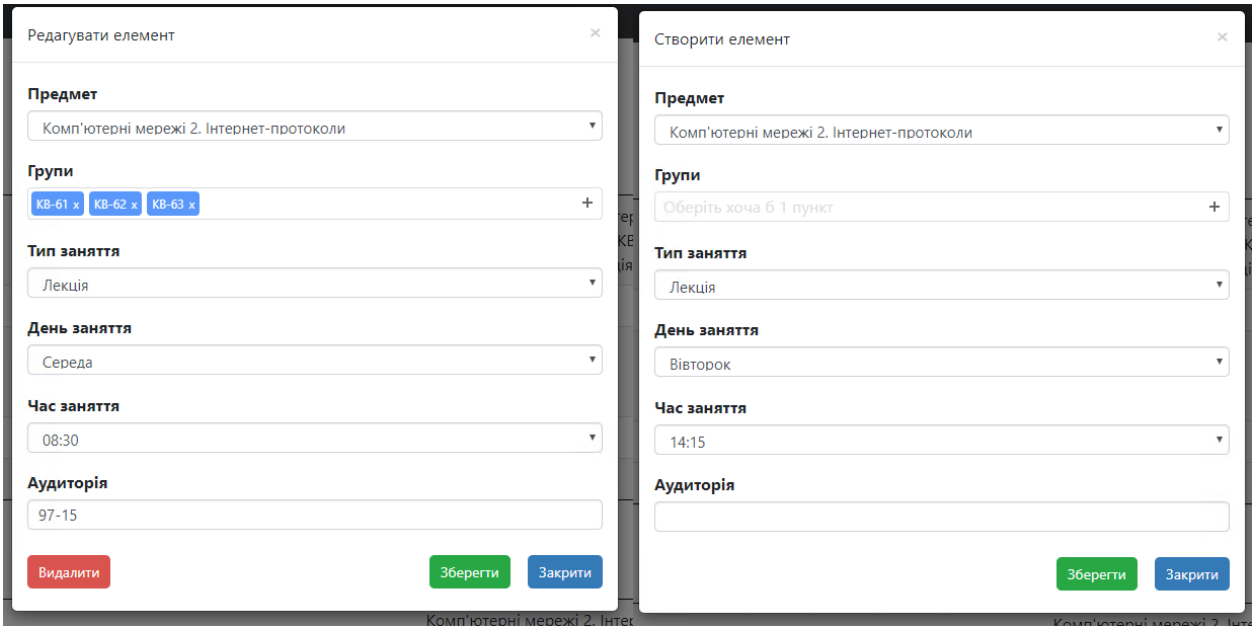


Рисунок 4.7 – Форми редагування та створення заняття

В них виводяться предмети, прив’язані за даним викладачем. Після підтвердження дії, розклад буде змінено.

Перейдемо до наступної вкладки – «Новини» (рисунок 4.8). Тут педагог може створювати новини для обраних груп за допомогою кнопки «Створити новину», а також редагувати і видаляти уже створені, за допомогою відповідних кнопок у таблиці.

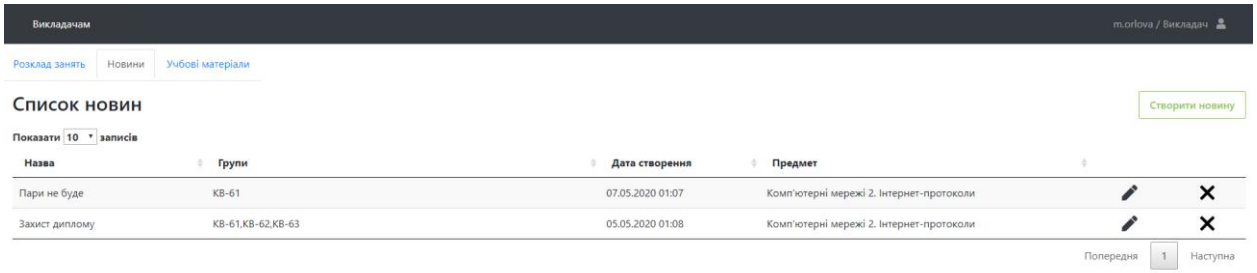


Рисунок 4.8 – вкладка «Новини»

Тут, як і у випадку з розкладом, також відкривається форма редагування (рисунок 4.9), після заповнення якої та подальшого збереження, усі студенти обраних груп зможуть її побачити.

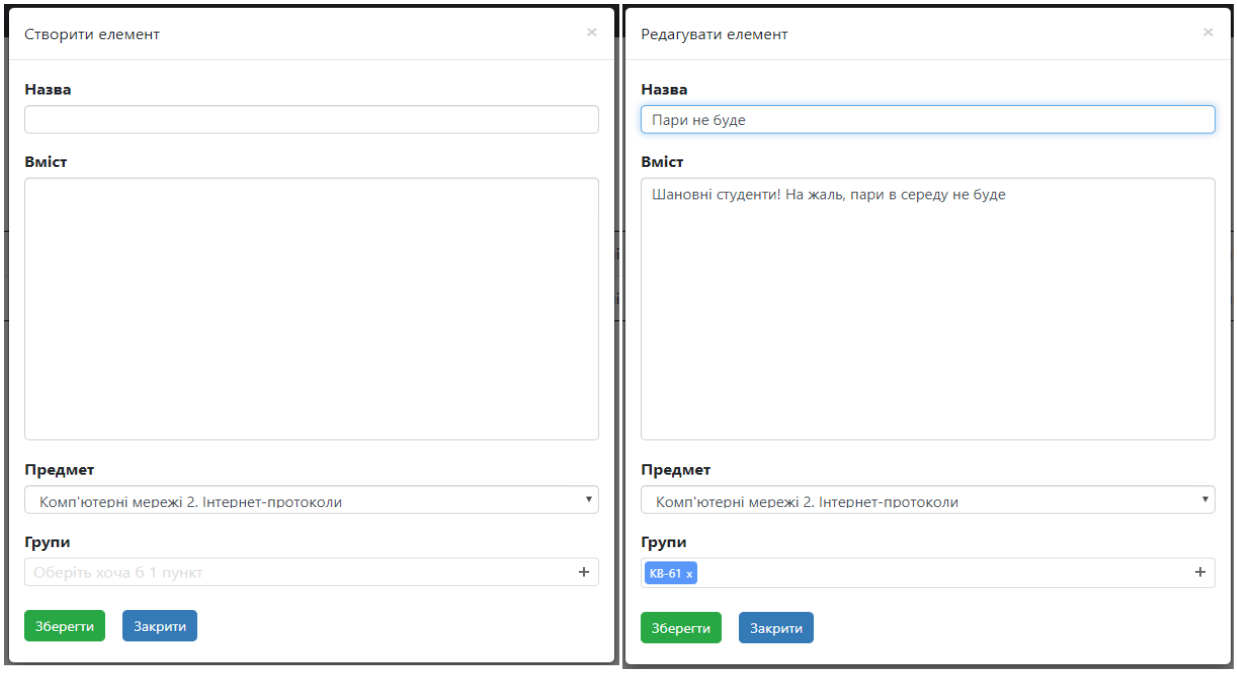


Рисунок 4.9 – Форми редагування і створення новими

Остання вкладка для розгляду – «Учбові матеріали» (рисунок 4.10). Тут викладач може завантажити будь-який текстовий файл на сервер, що потім зможуть отримати всі студенти обраних груп. Для прикладу, це можуть бути: додаткова література, семестрові бали, результати контрольної роботи тощо.

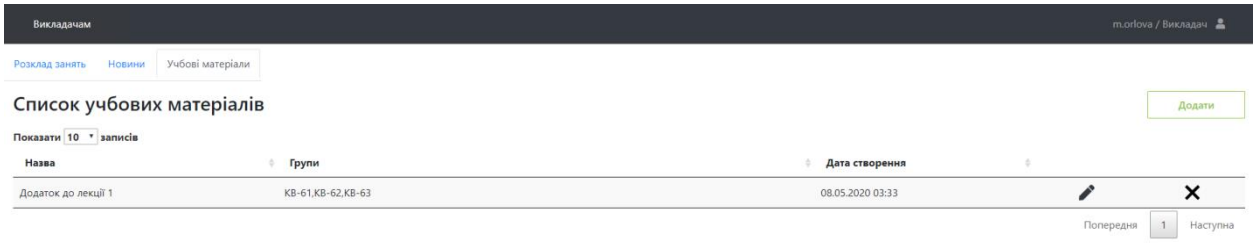


Рисунок 4.10 – Вкладка «Учбові матеріали»

Для завантаження файлу необхідно натиснути кнопку «Додати», яка відкриє форму створення файлу (рисунок 4.11). Після заповнення усіх полів, вибору файлу та підтвердження дії, матеріал буде завантажено на сервер.

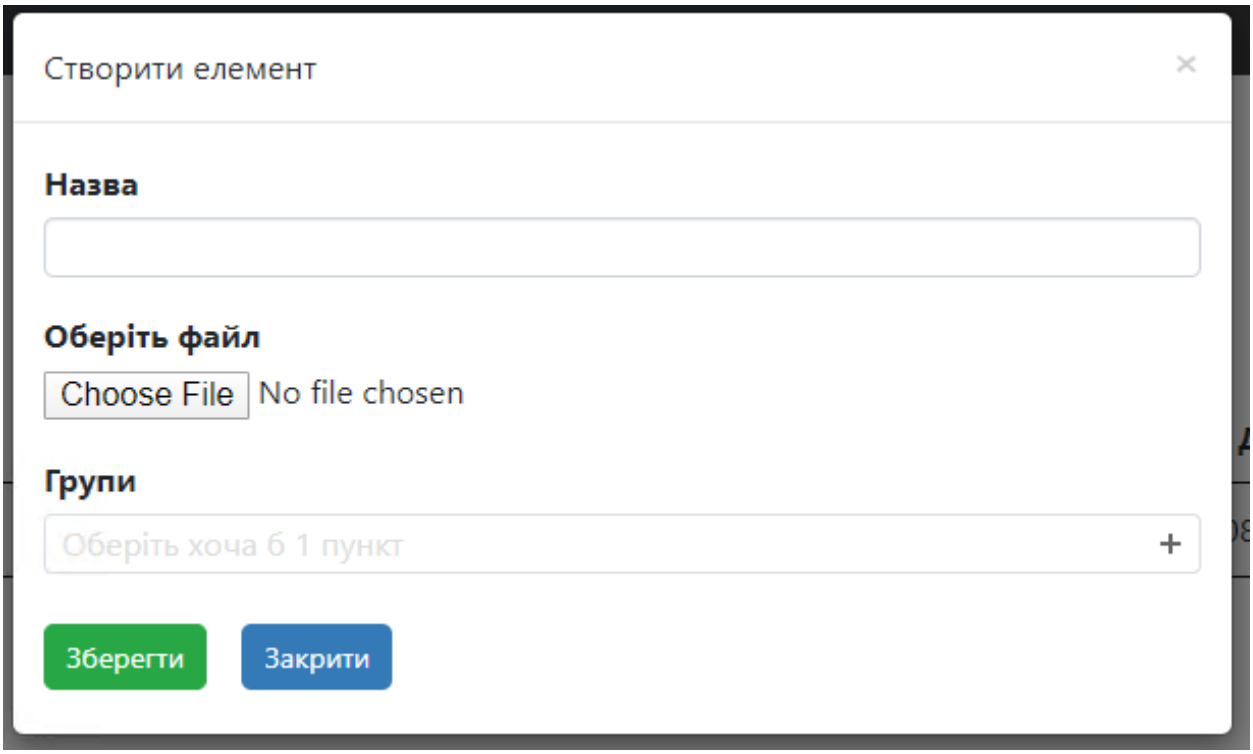


Рисунок 4.11 – Форма створення файлу

ВИСНОВКИ

Основною метою дипломної роботи була розробка університетського веб-порталу, який би вміщував усі необхідні засоби для процесу навчання. При цьому він був створений за допомогою сучасних технологій та має гнучку архітектуру для можливості розширення функціоналу в подальшому.

В ході написання проекту були проаналізовані засоби розробки веб-додатків, указані причини використання фреймворків з наступним вибором найбільш вдалого варіанту. Розглянуто особливості його складових. І нарешті, було обґрунтовано використання реляційної СУБД.

Розроблений проект розбитий на модулі, з детальним аналізом кожного з них. Демонстрація роботи додатку стала останнім пунктом дипломного проекту.

Портал має можливості для розвитку. Щодо нового функціоналу, то це може бути вкладка з розкладом сесії, де самі викладачі зможуть обирати дні екзаменів та консультацій. Гарним оновленням може бути введення системи сповіщень для вкладки «Новини».

У кінцевому підсумку проект показав свою спроможність для використання учбовим закладом з метою покращення процесу навчання.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. AngularJS documentation [Електронний ресурс] – 2015. Режим доступу: https://docs.angularjs.org/misc/faq. Дата доступу: травень 2020.
2. The PHP Framework for web artisants [Електронний ресурс] – 2019. Режим доступу: https://laravel.com/docs/7.x. Дата доступу: травень 2020.
3. Веб-портал розробника ASP.NET MVC 5 [Електронний ресурс] - 2018. – Режим доступу: https://docs.microsoft.com/en- us/aspnet/mvc/overview/ getting-started/introduction/getting-started. – Дата доступу: квітень 2020.
4. Michael B.White – «Mastering C# (C Sharp Programming)», 2019 [Книга]
5. Веб-портал розробника Microsoft SQL Server [Електронний ресурс] – 2019. Режим доступу: https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver15. Дата доступу: квітень 2020.
6. Erich Gamma, Richard Helm, Ralph Johnson, John Vilssdes – «Design Patterns», 1994 [Книга].